CATALOG

Bloc, Inc. Head Quarters
110 Sutter Street – 10th Floor
San Francisco, CA 94104
(404) 480-2562
www.bloc.io
No physical campus; distance learning only

TABLE OF CONTENTS:

MISSION

Our mission at Bloc is to use technology to provide world-class engineering and design education to anyone with the motivation to learn.

The U.S. economy is in the midst of transitioning to a tech-based economy, where even traditional industries like food, transportation, and finance are being transformed by technology. Venture capitalist Marc Andreessen calls this "software eating the world." To keep up with this accelerating pace of change, we need a system of education that can train students for the skills needed in a tech-based economy while also being affordable, accessible, and effective at delivering great student outcomes.

Bloc's programs are entirely online and structured around 1-on-1 mentorship. We borrow from the apprenticeship model of education where students learn 1-on-1 with their mentor as they go through project-based curriculum, with the goal of gaining real-world skills and a portfolio of demonstrated work. Our students spread the globe and choose Bloc because they are often unable to meet the requirements of traditional education that asks them to quit their jobs, relocate to a new city, and pay an inordinate amount of tuition to further their education. Bloc provides students with a modern education that would have otherwise been unavailable to them, and has transformed hundreds of lives along the way.

Our objectives at Bloc are:

- To be genuine and authentic student advocates, first and foremost.
- To provide a world class education to anyone with the determination to succeed.
- To utilize technology to continuously improve the accessibility and efficacy of our education.
- To celebrate and share our pride in craftsmanship, a core value we try to instill in our students as they start new careers in software engineering and design.

APPROVAL TO OPERATE

California
Bloc, Inc. is a private institution that has applied for approval to operate with the Bureau for Postsecondary Education (BPPE). BPPE is an agency responsible for granting authority to operate and provide oversight of California's private postsecondary educational institutions.

Bureau for Private Postsecondary Education 2535 Capitol Oaks Drive, Suite 400 Sacramento, California 95833 Phone: 916-431-6959
Fax: 916-2631897 Website: www.bppe.ca.gov

As a prospective student, you are encouraged to review this catalog prior to signing an enrollment agreement. You are also encouraged to review the School Performance Fact Sheet, which must be provided to you prior to signing an enrollment.

DISCLOSURE STATEMENT REGARDING BANKRUPTCY

Bloc, Inc. does not have a pending petition in bankruptcy, is not operating as a debtor in possession, has not filed a petition in bankruptcy within the preceding five years, and has not had a petition of bankruptcy filed against it within the preceding five years that resulted in reorganization under Chapter 11 of the United States Bankruptcy Code (11 U.S.C Sec. 1101, et seq.).

ADMISSIONS REQUIREMENTS

The following admissions policies apply to all Bloc programs:

- Students must be 18 years old or older to enroll at Bloc.
- Admission into any Bloc program requires that the student have a high school diploma or equivalent (General Education Diploma – GED) or a diploma from an institution of higher education accredited by an accrediting association recognized by the U.S. Department of Education. Bloc does not admit ability-to-benefit students.
- Bloc does not accept transfer credit, nor does Bloc offer challenge or achievement tests.
- Students must demonstrate operating proficiency on a computer at the time of enrollment.
- Bloc programs are provided in English only. Students from other countries where the primary language is not English must demonstrate operating proficiency of English prior to being accepted into the program. All instruction at Bloc will be conducted in English. Bloc does not offer or provide English language services, including instruction such as ESL.
- Bloc does not offer Visa services for foreign students, nor does Bloc vouch for student status.
- Students must possess basic reading and arithmetic skills at the time of enrollment.

ARTICULATION AGREEMENTS

Bloc has not entered into an articulation or transfer agreement with any other college or university.

GENERAL EDUCATION REQUIREMENTS

Bloc programs do not require students to complete general education courses as part of the curriculum.

ACCREDITATION

Bloc, Inc. is not accredited by an accreditation body recognized by the U.S. Department of Education. Bloc has not received a provisional approval and is not offering an unaccredited degree program.

PRIOR EXPERIENTIAL LEARNING

Bloc does not award credit for prior experiential learning.

GRADUATE LICENSURE

The goal of Bloc programs is not licensure and the profession, occupation, trade or career field from which Bloc equips graduates does not require licensure.

METHOD OF INSTRUCTION

The following method of instruction applies to all Bloc programs:

- Students read and watch curriculum available at bloc.io. Each program has several phases, and each phase has several checkpoints.
- Students complete exercises at the end of each checkpoint to be reviewed by their mentor.
- Students may exchange electronic correspondence with their mentor using either the messaging system on bloc.io or using email.
- Students meet with their mentor via video chat 1-3 times per week to "pair program" (write and review software code concurrently with a mentor), discuss the curriculum, and/or ask questions about projects in progress.
- Students are not required to submit thru the mail, and the expected response time via electronic submission is generally 24 hours.

NOTICE CONCERNING TRANSFERABILITY OF CREDITS AND CREDENTIALS EARNED AT OUR INSTITUTION

The transferability of credits you earn at Bloc, Inc. is at the complete discretion of an institution to which you may seek to transfer. Acceptance of the Certificate of Completion you earn in the educational program is also at the complete discretion of the institution to which you may seek to transfer. If the Certificate of Completion that you earn at this institution are not accepted at the institution to which you seek to transfer, you may be required to repeat some or all of your coursework at that institution. For this reason, you should make certain that your attendance at this institution will meet your educational goals. This may include contacting an institution to which you may seek to transfer after attending Bloc, Inc. to determine if your Certificate of Completion) will transfer.

FACILITIES AND EQUIPMENT

Bloc does not have any physical classrooms or locations, as its programs are entirely online. There is no physical campus.

Bloc programs require a computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.

LIBRARIES AND OTHER LEARNING RESOURCES

Bloc does not have a physical library or tangible learning resource repository. However, Bloc does maintain a list of free resources for students of several programs who wish to supplement the Bloc curriculum.

- Designer Track: https://www.bloc.io/resources/designer-track-prework
- Rails Fundamentals: https://www.bloc.io/resources/rails-prework
- Frontend Fundamentals: https://www.bloc.io/resources/frontend-prework
- iOS Fundamentals: https://www.bloc.io/resources/ios-prework
- Android Fundamentals: https://www.bloc.io/resources/android-prework

ATTENDANCE POLICY

Student understands and acknowledges that the program is very intense and requires consistent attendance and dedication. Students are required to attend the number of mentor sessions and complete the number of hours of study, per week, associated with the program they enroll in. The breakdown of mentor sessions, per program, and hours of study, per program, are noted in the "Description of Programs Offered" section listed below.

Students who fail to attend their mentor sessions or complete the required academic work for three consecutive weeks will be withdrawn from the program and a refund calculation conducted based on Bloc's Refund Policy.

LEAVE OF ABSENCE

If life happens — a serious illness, an unexpected life change — students have the option to freeze their program in one-week increments for a total of four weeks. Bloc will save their progress, and when they return they will pick-up where they left off. Bloc does not offer extended freezing so students should make sure to save this option for only when it's necessary.

PROBATION AND DISMISSAL POLICY

If a student fails to complete their program within two years of their program Start Date (three years if enrolled in the SET pace of 108-weeks, and four years if enrolled in the SET pace of 144-weeks), they will be withdrawn from the program.

If a student fails to complete portions of the program and/or does not correspond with Bloc for three consecutive weeks, Bloc will withdraw the student and issue a refund pursuant to Bloc's

Refund Policy. In this situation, the withdrawal date will be in keeping with Bloc's Cancellation and Refunds policy.

Bloc does not have a probation status or policy.

HOUSING

Bloc's programs are offered entirely via distance education to students across the country. Bloc does not provide student housing services or dormitory facilities, because students participate in Bloc's programs from their own homes.

DESCRIPTION OF PROGRAMS OFFERED

Programs Offered:

- Software Developer Track
  - A comprehensive program to provide graduates with the skills required to work as a Software Engineer at a technology company
  - Students may select one of three paces on which to complete this program:
    - Fast: 40 hours of study and 3 mentor meetings each week to finish in 36 weeks
    - Moderate: 25 hours of study and 2 mentor meetings each week to finish in 54 weeks
    - Slow: 15 hours of study and 1 mentor meeting each week to finish in 108 weeks
  - Students meeting graduation requirements earn a Certificate of Completion
  - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- Web Developer Track
  - Provide graduates with the skills required to work as a junior web developer
  - Students may select one of two paces on which to complete this program:
    - Moderate: 20 hours of study and 2 mentor meetings each week to finish in 27 weeks
    - Slow: 12 hours of study and 1 mentor meeting each week to finish in 54 weeks
  - Students meeting graduation requirements earn a Certificate of Completion
  - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- Designer Track
  - Provide graduates with the skills required to work as a digital designer
  - Students may select one of three paces on which to complete this program:

        - Fast: 40 hours of study and 3 mentor meetings each week to finish in 24 weeks
        - Moderate: 25 hours of study and 2 mentor meetings each week to finish in 36 weeks
        - Slow: 15 hours of study and 1 mentor meeting each week to finish in 72 weeks
    - Students meeting graduation requirements earn a Certificate of Completion
    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- Rails Fundamentals
    - Provide graduates with the ability to build basic applications using the Rails web development framework
    - Students may select one of two paces on which to complete this program:
        - Moderate: 12 hours of study and 2 mentor meetings each week to finish in 16 weeks
        - Slow: 25 hours of study and 1 mentor meeting each week to finish in 32 weeks
    - Students meeting graduation requirements earn a Certificate of Completion
    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- Frontend Fundamentals
    - Provide graduates with the ability to build interactive web pages using HTML, CSS, Javascript and Angular
    - Students may select one of two paces on which to complete this program:
        - Moderate: 12 hours of study and 2 mentor meetings each week to finish in 16 weeks
        - Slow: 25 hours of study and 1 mentor meeting each week to finish in 32 weeks
    - Students meeting graduation requirements earn a Certificate of Completion
    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- iOS Fundamentals
    - Provide graduates with the ability to build basic mobile applications for iOS devices
    - Students may select one of two paces on which to complete this program:
        - Moderate: 12 hours of study and 2 mentor meetings each week to finish in 16 weeks
        - Slow: 25 hours of study and 1 mentor meeting each week to finish in 32 weeks
    - Students meeting graduation requirements earn a Certificate of Completion

    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- Android Fundamentals
    - Provide graduates with the ability to build basic mobile applications for Android devices
    - Students may select one of two paces on which to complete this program:
        - Moderate: 12 hours of study and 2 mentor meetings each week to finish in 16 weeks
        - Slow: 25 hours of study and 1 mentor meeting each week to finish in 32 weeks
    - Students meeting graduation requirements earn a Certificate of Completion
    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.
- UX/UI Design Fundamentals
    - Provide graduates with the ability to build functional interfaces for web and mobile applications
    - Students may select one of two paces on which to complete this program:
        - Moderate: 12 hours of study and 2 mentor meetings each week to finish in 16 weeks
        - Slow: 25 hours of study and 1 mentor meeting each week to finish in 32 weeks
    - Students meeting graduation requirements earn a Certificate of Completion
    - Equipment Required: Computer with a microphone and speakers, and high-speed internet access. Bloc does not provide computers to students, and every student must own or have access to a personal computer at least 4GB RAM, at least 2GHz, and at least 100 GB HD.

Software Developer Track

Phase One: Programming Fundamentals

Learn HTML, CSS, the Command Line, Git, GitHub, Ruby, and JavaScript. You'll use all of these tools extensively throughout Bloc as you build projects.

- Build and style web pages with HTML and CSS
- Master your terminal and file system with Command Line operations in Unix
- Manage your code with Git and GitHub
- Code frontend programs with JavaScript
- Code backend programs with Ruby
- Master object-oriented programming techniques and best-practices with JavaScript and Ruby

Phase Two: Foundations of Web Development

Apply your programming skills to web development as you build four projects. There are two parts of this phase: Frontend Web Development with JavaScript and Backend Web Development with Ruby.

FRONTEND WEB DEVELOPMENT

Apply HTML and CSS to Web Applications

- Build your first application, a replica of Spotify
- Write semantic HTML
- Use HTML5 data attributes
- Use best-practice CSS architecture
- Include external assets like images and fonts
- Create responsive websites with media queries
- Use CSS transitions and animations

Write Web Applications in JavaScript

- Use JavaScript primitives, operators, arrays, and conditionals
- Apply object-oriented JavaScript with constructors and prototypes
- Use functions, scopes, and closures for maintainable and efficient code

Master the DOM

- Learn about the Document Object Model (DOM) and its capabilities
- Use different types of DOM nodes and selectors
- Add and remove content with DOM Scripting
- Understand browser events like propagation, handling, event delegation, and event objects

Frontend Project: Build a Portfolio

- Create a unique Portfolio site that showcases your personality, projects, and writing
- Include images and links for your projects
- Build a blogging engine to explain your coding process and express your opinions
- Make your site stand out with a stunning design

Frontend Project: Angular

- Explore one of the most popular JavaScript MVC frameworks
- Learn about Angular Controllers, Directives, Services and Modules
- Refactor Bloc Jams to work with Angular

Frontend Project: Bloc Chat

- Build a replica of Slack - a real-time chat application
- Use Grunt to build and serve your assets
- Learn Firebase, a backend as a service (BaaS) to sync messages in real-time
- Learn how to use an external API with Angular, Firebase's AngularFire
- Use cookies to let users set a screen name
- Create multiple chat rooms for different subjects

BACKEND WEB DEVELOPMENT

- Use the Ruby on Rails framework to build full-stack web applications
- Employ test-driven development to write bug-resistant code
- Apply object-oriented best-practices to build maintainable and performant web applications
- Build Ruby gem libraries for other developers to use in their applications
- Design and build APIs to allow different applications to communicate with each other

Ruby Command Line Project: Build an Address Book

- Build a command line address book application using Test-Driven Development
- Learn how to run a command line application
- Build data models with Ruby classes
- Build a menu system to process user input
- Import data from text files
- Learn about algorithms by building a binary search feature

Rails Project: Build a Replica of Reddit

Master the basics:

- Learn the fundamentals of web development with Rails
- Learn the fundamentals of frontend markup with HTML and CSS
- Leverage Rails' API, including ActiveRecord, ActionView, ActionController, and ActionMailer
- Learn the fundamentals of database architecture, design, and maintenance
- Employ database best practices, including indexing, efficient querying, and scope-chaining
- Build robust features from scratch, including user authentication, authorization, and data-seeding

Learn intermediate programming techniques:

- Learn best practices for debugging and troubleshooting
- Write validation tests to ensure data integrity
- Learn how to integrate packaged solutions for markdown rendering, pagination, and image management
- Learn how to create complex routes and associations

Build advanced features:

- Write a time-decay algorithm for voting and ranking
- Build a "Favoriting" feature using after-action callbacks and automatic emails
- Authorize private topics and public profiles

Backend Project: Build a Ruby Gem using an API

- Ruby Gems are plug-ins that you can add to an existing Rails app. This project will teach you to build a Ruby Gem of your own
- Explore authorization with JSON Web Tokens and understand the reasons to call an API using a client
- Explain how to update resources via an API

Backend Project: Software-as-a-Service Wiki Collaboration Tool

- Build an application to allow users to create, collaborate, and share wikis
- Integrate Stripe to charge users for premium accounts with additional features

Phase Three: Specialization

You'll work with your mentor and our Outcomes team to select at least two more projects based on your outcome goal. These projects will stand out in your portfolio and position you for interviews with companies.

- Use new JavaScript frameworks
- Experiment with different database technologies
- Build voice-activated software using Amazon's Alexa platform
- Implement payment systems for software-as-a-service (SaaS) applications
- Build or contribute to open-source projects
- Create game applications
- Bring your own idea to life with the Capstone project

Here are some examples of projects that students have built in the Specialization Phase:
Self-Destructing Todo List

- Build a to-do list application that deletes items automatically after a given amount of time
- Learn how to code and schedule automated tasks with Rake
- Implement robust authentication and authorization for different types of users

Social Bookmarking Tool - Build an application to allow users to share their favorite links - Learn how to send and receive email automatically

API Analytics Service - Build an analytics service for tracking application users and their activity - Use JavaScript, jQuery, JSON, and AJAX to send tracking events - Build a server-side API to persist tracked events - Create a dashboard report for viewing user activity

To-Do List as-a-Service - Build an open API for a to-do list application - Use Rails Serializers to format data into API-friendly JSON - Learn best practices for designing

RESTful APIs - Learn about internet security, CSRF, and how you can use Rails to protect your users

Analytics as a Service

- Learn how to build an Analytics API that tracks metrics like most popular songs in your application
- Visualize the Data using D3 or ChartJS
- Learn about the powerful HTML5 Canvas element for visualizing your data
- Create a full-featured dashboard that tracks your events in real-time

Pong

- Create the game logic using JavaScript functions and objects, then create the visualization
- Use JavaScript's requestAnimationFrame to create seamless animation graphics
- Explore another use-case for the HTML5 Canvas Element
- Learn how to use physics in your motion graphics by adjusting things like speed and x, y positioning (we promise, not too much math!)

BlocTime

- BlocTime loosely emulates the Pomodoro time management technique to manage day-to-day tasks
- Use Grunt to build and serve your assets
- Use the Firebase API for persistent storage of the tasks you've completed

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Deploy your application, and we'll help you promote it

Career Support

The Bloc Career Support Program is a holistic collection of curriculum and services designed to prepare students for the technical recruiting process and conduct a successful job search. The Part-Time Web Developer Track roadmap includes dedicated material to review with an experienced mentor in preparation for the recruiting process. Programming Reinforcement exercises challenge you to problem solving with data structure and algorithms to help you master the whiteboard interview.

- Bloc Web Developer Track students receive dedicated support prior to and throughout their first technical job search, including resume and portfolio critique and a review of LinkedIn and GitHub profiles to ensure the best possible presentation to prospective employers.
- Mentors lead mock technical interviews so students can handle real technical interviews with confidence.

- Define criteria to guide their job search, and implement a process and cadence for managing the search.
- Access to recorded Career Talks and TechTalks featuring top employers like Amazon, Mozilla, and Twilio

Phase Four: Software Engineering Principles

A software engineer understands how and why something works, not just how to use it. They can architect solutions, understand trade-offs, and can account for scale and change. Software engineers understand computer science, and apply it to solve complex problems efficiently.

DATA STRUCTURES
- Stacks
- Queues
- Linked Lists
- Hashes
- Binary Trees
- K-Ary Trees
- Graphs
- Calculate the impact of using different data structures

ALGORITHMS AND COMPLEXITY ANALYSIS
- Searching
- Sorting
- Complexity Analysis
- Complexity Improvement
- Calculate the impact of using different algorithms
- Analyze the complexity of algorithms

DATABASES AND ADVANCED SQL
- Database schema design
- Advanced SQL statements and querying patterns
- Object-relational mapping patterns
- Build an ORM layer for a database

FRAMEWORK ARCHITECTURE AND DESIGN PATTERNS
- Web framework design
- RESTful API design
- Build a full-stack web development framework
- Optimize software performance

Phase Five: Open-Source Apprenticeship

- Build your own open-source software
- Contribute to real open-source software projects

- Gain experience collaborating with professional engineers
- Learn to write maintainable and performant code
- Learn the best-practices of open-source development
- Build a unique portfolio of open-source projects and contributions that will be publicly visible on your GitHub profile

Career Support Program

The Bloc Career Support Program is a holistic collection of curriculum and services designed to prepare students for the technical recruiting process and conduct a successful job search.

The Software Engineering Track roadmap includes dedicated material to review with an experienced mentor in preparation for the recruiting process. Programming Reinforcement exercises challenge you to problem solving with data structure and algorithms to help you master the whiteboard interview.

- Students receive dedicated support prior to and throughout their first technical job search, including resume and portfolio critique and a review of LinkedIn and GitHub profiles to ensure the best possible presentation to prospective employers.
- Mentors lead mock technical interviews so students can handle real technical interviews with confidence.
- Define criteria to guide their job search, and implement a process and cadence for managing the search.
- Access to recorded Career Talks and TechTalks featuring top employers like Amazon, Mozilla, and Twilio

Web Developer Track

Phase One: Programming Fundamentals

Learn HTML, CSS, the Command Line, Git, GitHub, Ruby, and JavaScript. You'll use all of these tools extensively throughout Bloc as you build projects.

- Build and style web pages with HTML and CSS
- Master your terminal and file system with Command Line operations in Unix
- Manage your code with Git and GitHub
- Code frontend programs with JavaScript
- Code backend programs with Ruby
- Master object-oriented programming techniques and best-practices with JavaScript and Ruby

Phase Two: Foundations of Web Development

Apply your programming skills to web development as you build four projects. There are two parts of this phase: Frontend Web Development with JavaScript and Backend Web Development with Ruby.

FRONTEND WEB DEVELOPMENT

Apply HTML and CSS to Web Applications

- Build your first application, a replica of Spotify
- Write semantic HTML
- Use HTML5 data attributes
- Use best-practice CSS architecture
- Include external assets like images and fonts
- Create responsive websites with media queries
- Use CSS transitions and animations

Write Web Applications in JavaScript

- Use JavaScript primitives, operators, arrays, and conditionals
- Apply object-oriented JavaScript with constructors and prototypes
- Use functions, scopes, and closures for maintainable and efficient code

Master the DOM

- Learn about the Document Object Model (DOM) and its capabilities
- Use different types of DOM nodes and selectors
- Add and remove content with DOM Scripting
- Understand browser events like propagation, handling, event delegation, and event objects

Frontend Project: Build a Portfolio

- Create a unique Portfolio site that showcases your personality, projects, and writing
- Include images and links for your projects
- Build a blogging engine to explain your coding process and express your opinions
- Make your site stand out with a stunning design

Frontend Project: Angular

- Explore one of the most popular JavaScript MVC frameworks
- Learn about Angular Controllers, Directives, Services and Modules
- Refactor Bloc Jams to work with Angular

Frontend Project: Bloc Chat

- Build a replica of Slack - a real-time chat application
- Use Grunt to build and serve your assets
- Learn Firebase, a backend as a service (BaaS) to sync messages in real-time
- Learn how to use an external API with Angular, Firebase's AngularFire
- Use cookies to let users set a screen name

- Create multiple chat rooms for different subjects

BACKEND WEB DEVELOPMENT

- Use the Ruby on Rails framework to build full-stack web applications
- Employ test-driven development to write bug-resistant code
- Apply object-oriented best-practices to build maintainable and performant web applications
- Build Ruby gem libraries for other developers to use in their applications
- Design and build APIs to allow different applications to communicate with each other

Ruby Command Line Project: Build an Address Book

- Build a command line address book application using Test-Driven Development
- Learn how to run a command line application
- Build data models with Ruby classes
- Build a menu system to process user input
- Import data from text files
- Learn about algorithms by building a binary search feature

Rails Project: Build a Replica of Reddit

Master the basics:

- Learn the fundamentals of web development with Rails
- Learn the fundamentals of frontend markup with HTML and CSS
- Leverage Rails' API, including ActiveRecord, ActionView, ActionController, and ActionMailer
- Learn the fundamentals of database architecture, design, and maintenance
- Employ database best practices, including indexing, efficient querying, and scope-chaining
- Build robust features from scratch, including user authentication, authorization, and data-seeding

Learn intermediate programming techniques:

- Learn best practices for debugging and troubleshooting
- Write validation tests to ensure data integrity
- Learn how to integrate packaged solutions for markdown rendering, pagination, and image management
- Learn how to create complex routes and associations

Build advanced features:

- Write a time-decay algorithm for voting and ranking
- Build a "Favoriting" feature using after-action callbacks and automatic emails
- Authorize private topics and public profiles

Backend Project: Build a Ruby Gem using an API

- Ruby Gems are plug-ins that you can add to an existing Rails app. This project will teach you to build a Ruby Gem of your own
- Explore authorization with JSON Web Tokens and understand the reasons to call an API using a client
- Explain how to update resources via an API

Backend Project: Software-as-a-Service Wiki Collaboration Tool

- Build an application to allow users to create, collaborate, and share wikis
- Integrate Stripe to charge users for premium accounts with additional features

Phase Three: Specialization

You'll work with your mentor and our Outcomes team to select at least two more projects based on your outcome goal. These projects will stand out in your portfolio and position you for interviews with companies.

- Use new JavaScript frameworks
- Experiment with different database technologies
- Build voice-activated software using Amazon's Alexa platform
- Implement payment systems for software-as-a-service (SaaS) applications
- Build or contribute to open-source projects
- Create game applications
- Bring your own idea to life with the Capstone project

Here are some examples of projects that students have built in the Specialization Phase:

Self-Destructing Todo List

- Build a to-do list application that deletes items automatically after a given amount of time
- Learn how to code and schedule automated tasks with Rake
- Implement robust authentication and authorization for different types of users

Social Bookmarking Tool - Build an application to allow users to share their favorite links - Learn how to send and receive email automatically

API Analytics Service - Build an analytics service for tracking application users and their activity - Use JavaScript, jQuery, JSON, and AJAX to send tracking events - Build a server-side API to persist tracked events - Create a dashboard report for viewing user activity

To-Do List as-a-Service - Build an open API for a to-do list application - Use Rails Serializers to format data into API-friendly JSON - Learn best practices for designing RESTful APIs - Learn about internet security, CSRF, and how you can use Rails to protect your users

Analytics as a Service

- Learn how to build an Analytics API that tracks metrics like most popular songs in your application
- Visualize the Data using D3 or ChartJS

- Learn about the powerful HTML5 Canvas element for visualizing your data
- Create a full-featured dashboard that tracks your events in real-time

Pong

- Create the game logic using JavaScript functions and objects, then create the visualization
- Use JavaScript's requestAnimationFrame to create seamless animation graphics
- Explore another use-case for the HTML5 Canvas Element
- Learn how to use physics in your motion graphics by adjusting things like speed and x, y positioning (we promise, not too much math!)

BlocTime

- BlocTime loosely emulates the Pomodoro time management technique to manage day-to-day tasks
- Use Grunt to build and serve your assets
- Use the Firebase API for persistent storage of the tasks you've completed

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Deploy your application, and we'll help you promote it

Career Support

The Bloc Career Support Program is a holistic collection of curriculum and services designed to prepare students for the technical recruiting process and conduct a successful job search.
The Part-Time Web Developer Track roadmap includes dedicated material to review with an experienced mentor in preparation for the recruiting process. Programming Reinforcement exercises challenge you to problem solving with data structure and algorithms to help you master the whiteboard interview.

- Bloc Web Developer Track students receive dedicated support prior to and throughout their first technical job search, including resume and portfolio critique and a review of LinkedIn and GitHub profiles to ensure the best possible presentation to prospective employers.
- Mentors lead mock technical interviews so students can handle real technical interviews with confidence.
- Define criteria to guide their job search, and implement a process and cadence for managing the search.
- Access to recorded Career Talks and TechTalks featuring top employers like Amazon, Mozilla, and Twilio

Designer Track

Pre-Work (Complete during orientation week)

- Introduction to Adobe Photoshop (4 hours)
- Introduction to Adobe Illustrator (8 hours)
- Getting Started with Sketch (2 hours)
- HTML and CSS Fundamentals (7 hours)
- Build Four Webpages from Scratch (4 hours)
- GitHub Tutorial (15 mins)
- Assessment - submit two assignments to your lead mentor (1 hour)

Intro to Bloc

- A 1-on-1 orientation with one of Bloc's Student Advisors
- Intro meeting with your Mentor, and access to a Hacker Team
- The Project-Based Approach to Learning
- Learn new concepts and apply them to design your first web application

Learn the Tools & Fundamentals

- Learn how to use design tools like Photoshop, Illustrator, & Sketch
- Learn design fundamentals and how to properly use typography, layout, color, & branding in your projects
- Follow best practices to solve design problems & understand how users interact with web/mobile applications
- Learn about user-centered design, and apply design research methods to focus on user goals
- Learn how personas and user research can influence your final product
- Learn the importance of information architecture and content strategy
- Learn the soft-skills of design: presenting, selling, defending and critiquing your work

Design Deep Dive

- Complete the full-scale design process for your first web app by using the fundamentals you learned and research you conducted
- Create wireframes, prototype, test, and iterate on your design decisions
- Prepare high-fidelity mockups for development, and measure success through usability testing

Mobile Design

- Learn how web design translates to mobile app design patterns
- Learn to design apps for both of the dominant mobile platforms, iOS and Android
- Design replicas of Spotify for iPhone and WhatsApp for Android

Configure Your Developer Environment

- Basic Command-Line operations
- Git and GitHub
- Learn to use your text editor

Introduction to HTML, CSS & Sass

- Learn Frontend Development building your first app
- Semantic HTML
- Including CSS in an HTML Page
- Including external assets like images and fonts
- Responsive CSS with Media Queries
- Building a responsive grid system
- Learn how to write Sass

Introduction to JavaScript & jQuery

- Learn the basics of JavaScript
- Learn about the Document Object Model (DOM) and its capabilities
- Including External Libraries like jQuery

Intro to Project Phase

- Work on four to six open-ended projects and create a portfolio
- Your mentor will act as a client or senior designer
- Your mentor will provide you with project requirements, and help you to apply what you've learned to design new apps from scratch
- Work with your mentor to plan the best approach, critique your work, pair design, and revise

Paycrave

- Explore iOS and Android app design guidelines
- Create a mobile prototype that allows users to discover local food trucks
- Design user flows and the UI for a mobile payment solution

BlocShop

- Design an eCommerce shopping experience
- Create a responsive design for mobile, tablet and desktop devices
- Discover best practices for capturing information and displaying user-generated content

BlocStarter

- Design a complex web app that explores crowdfunding for charities and foundations

- Focus on a variety of users, creating personas, user flows, and wireframes
- Create an interactive prototype and test with real users to gather feedback

Product Showcase

- Design a marketing landing page for a specific product
- Focus on the principles of design, value proposition, and competitive differentiators
- Create a responsive web page

Capstone

- Design your own project
- Ship it with the expert guidance from your mentor
- Define the project scope and design it

Portfolio

- Create a personal website to showcase your work to future clients and employers
- Make it unique by defining your personal brand identity
- Design your portfolio site with a homepage, contact page, resume, and more

Rails Fundamentals

BUILD A DEVELOPMENT ENVIRONMENT

- Learn basic command line operations
- Become proficient with a code editor
- Learn how to manage code and projects with Git and GitHub
- Configure a Rails development stack
- Learn how to deploy web applications to a Production environment

RUBY FUNDAMENTALS

- Learn the fundamentals of object-oriented programming with Ruby
- Master the basics of the Ruby language, including strings, numbers, booleans, arrays, and conditional logic
- Practice intermediate aspects of Ruby, including methods, hashes, blocks, and loops
- Master object-oriented design basics with classes and modules
- Practice methodical debugging strategies
- Learn Test-Driven Development with the RSpec library

RUBY COMMAND LINE PROJECT: BUILD AN ADDRESS BOOK

- Build a command line address book application using Test-Driven Development
- Learn how to run a command line application
- Build data models with Ruby classes
- Build a menu system to process user input
- Import data from text files

- Learn about algorithms by building a binary search feature

YOUR FIRST RAILS APPLICATION: BUILD A REPLICA OF REDDIT

Master the basics:

- Learn the fundamentals of web development with Rails
- Learn the fundamentals of frontend markup with HTML and CSS
- Leverage Rails' API, including ActiveRecord, ActionView, ActionController, and ActionMailer
- Learn the fundamentals of database architecture, design, and maintenance
- Employ database best practices, including indexing, efficient querying, and scope-chaining
- Build robust features from scratch, including user authentication, authorization, and data-seeding

Learn intermediate programming techniques:

- Learn best practices for debugging and troubleshooting
- Write validation tests to ensure data integrity
- Learn how to integrate packaged solutions for markdown rendering, pagination, and image management
- Learn how to create complex routes and associations

Build advanced features:

- Write a time-decay algorithm for voting and ranking
- Build a "Favoriting" feature using after-action callbacks and automatic emails
- Authorize private topics and public profiles
- Learn advanced Test-Driven Development practices with RSpec
- Apply JavaScript and jQuery to manipulate data through AJAX
- Build a secure API to turn an application into a platform

TECHNICAL PROJECTS

You will complete at least two projects during the Project Phase. Projects are prescriptive yet open to interpretation and creativity. They are designed to challenge you while providing guidance to keep you on track. A project is complete when all its requirements are implemented and approved by your mentor.

Self-Destructing Todo List

- Build a to-do list application that deletes items automatically after a given amount of time
- Learn how to code and schedule automated tasks with Rake
- Implement robust authentication and authorization for different types of users

Software-as-a-Service Wiki Collaboration Tool

- Build an application to allow users to create, collaborate, and share wikis
- Integrate Stripe to charge users for premium accounts with additional features

Social Bookmarking Tool

- Build an application to allow users to share their favorite links
- Learn how to send and receive email automatically

API Analytics Service

- Build an analytics service for tracking application users and their activity
- Use JavaScript, jQuery, JSON, and AJAX to send tracking events
- Build a server-side API to persist tracked events
- Create a dashboard report for viewing user activity

To-Do List as-a-Service

- Build an open API for a to-do list application
- Use Rails Serializers to format data into API-friendly JSON
- Learn best practices for designing RESTful APIs
- Learn about internet security, CSRF, and how you can use Rails to protect your users

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Deploy your application

Frontend Fundamentals

FOUNDATION PHASE
Build your first application, a browser-based Spotify clone called Bloc Jams, while learning the fundamentals of frontend web development.

Configure Your Developer Environment

- Basic command line operations
- Git and GitHub
- Learn to use Brackets, your text editor

Introduction to HTML and CSS

- Learn Frontend Development building your first app: Bloc Jams
- Semantic HTML
- Including CSS in an HTML Page
- Including external assets like images and fonts
- Responsive CSS with Media Queries
- Building a responsive grid system

- Floats and Clearfixes

Introduction to JavaScript as a Programming Language

- JavaScript basics: Primitives, Operators, Arrays, and Conditionals
- Object-oriented JavaScript: Constructors, Prototypes, this, apply, call, and bind
- Functions, Scope, and Closures

JavaScript in the Browser

- CSS transitions and animations
- Include JavaScript in a web page
- Learn about the Document Object Model (DOM) and its capabilities
- The different types of DOM nodes
- DOM selectors
- Adding and removing content with DOM Scripting
- Browser events: propagation and handling
- JavaScript callbacks
- Event delegation and event objects
- HTML5 data attributes

jQuery

- Including External Libraries like jQuery
- Refactor DOM Scripting with jQuery
- Using jQuery events and helper functions
- Using the Buzz Audio library to play music in Bloc Jams

PROJECTS PHASE
You will complete at least two projects during the Project Phase. Projects are prescriptive yet open to interpretation and creativity, and designed to challenge you while providing guidance. A project is complete when all its requirements are implemented and approved by your mentor.

AngularJS

- Explore one of the most popular JavaScript MVC frameworks
- Learn about Angular Controllers, Directives, Services and Modules
- Refactor Bloc Jams to work with Angular

Bloc Chat

- Build a real-time chat application
- Use Grunt to build and serve your assets
- Learn Firebase, a backend as a service (BaaS) to sync messages in real-time
- Learn how to use an external API with Angular, Firebase's AngularFire
- Use cookies to let users set a screen name

- Create multiple chat rooms for different subjects

Bloc Jams Analytics

- Learn how to build an Analytics API that tracks metrics like most popular songs in your application
- Visualize the Data using D3 or ChartJS
- Learn about the powerful HTML5 Canvas element for visualizing your data
- Create a full-featured dashboard that tracks your events in real-time

Pong

- Create the game logic using JavaScript functions and objects, then create the visualization.
- Use JavaScript's requestAnimationFrame to create seamless animation graphics
- Explore another use-case for the HTML5 Canvas Element
- Learn how to use physics in your motion graphics by adjusting things like speed and x, y positioning (we promise, not too much math!)

BlocTime

- BlocTime loosely emulates the Pomodoro time management technique to manage day-to-day tasks
- Use Grunt to build and serve your assets
- Use the Firebase API for persistent storage of the tasks you've completed

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Deploy your application

iOS Fundamentals

BUILD A DEVELOPMENT ENVIRONMENT

- Learn how to use Xcode proficiently
- Master version control and code management with Git and GitHub
- Learn basic iOS controls like buttons, labels, and text fields

IOS FUNDAMENTALS: OBJECTIVE-C

- Basic Objective-C Syntax
- Numbers, variables, and strings
- If / else statements, different types of equality

- Loops, arrays, dictionaries, simple data types vs. objects
- Properties and scope

IOS FUNDAMENTALS: SWIFT

- Introduction to Swift
- Compare basics in Swift and Objective-C: properties, strings, arrays, dictionaries, loops, functions
- New concepts in Swift like tuples and closures

BEGINNER IOS APP: BUILD A CALCULATOR

Your first simple iOS app is an alcohol calculator that converts alcohol content for beer, whiskey, and wine.

- Learn how to build an app with storyboards, outlets, and actions
- Understand the impacts of proper sizing and styling with the UIViewController
- Learn basic navigation principles with the UINavigationController
- Build tabbed navigation with the UITabBarController

INTERMEDIATE IOS APP: BUILD A WEB BROWSER

Your second iOS app is a web browser with a colorful, floating toolbar. The web browser enhances your privacy by purging all browser history when you switch apps.

- Build a web browser app using WKWebView and delegation
- Learn how to clear browser history with UIApplicationDelegate
- Add a new toolbar by subclassing UIView and incorporating touch events
- Learn to recognize gesture events in your app
- Move and resize a floating toolbar using gesture recognizers
- Use gesture recognizers to respond to different touch patterns and respond accordingly

ADVANCED IOS APP: BUILD A REPLICA OF INSTAGRAM

Master the basics: - Learn how to display a dynamic list of images in a feed - Crate user accounts and display usernames and captions - Implement infinite scroll and pull-to-refresh - Learn how to leverage an API and connect to Instagram to display your friends' actual photos, captions, and comments - Understand data persistence with Keychain - remember images and login between launches

Apply intermediate programming techniques: - Add full-screen photo viewing and saving (custom UIViewController transitions) - Build a fancy "like" button (intro to Core Animation) - Post images to Instagram (interacting with other apps) - Customize your app for iPad
Build advanced features and prepare for deployment: - Write unit tests - What is Test-Driven Development? - Check for logic errors (static analysis) - Install analytics and crash handling services - Set your app icons and launch images - Create your app on iTunes Connect

TECHNICAL PROJECTS

You will complete at least two projects during the Project Phase. Projects are prescriptive yet open to interpretation and creativity. They are designed to challenge you while providing guidance to keep you on track. A project is complete when all its requirements are implemented and approved by your mentor.

Blocstagram Watch

- Build upon the Instagram app you created in the Foundation Phase to add an Apple Watch extension.
- Translate your app from Objective-C to Swift

Pong Replica

- Build a working replica of Pong
- Learn how to use UIKit Dynamics; a physics-based animation engine that's used for moving around basic views
- Add cool features like a bonus mode that catapults your Pong ball (inspired by AngryBirds!) so that you learn to use powerful physics
- Add Artificial Intelligence to your pong app so that you can play against the computer and crush its robotic spirit

To-Do List App (Evernote Replica)

- Build a note-taking app that syncs with iCloud
- Learn to work with Core Data - a powerful Apple framework for storing information and relationships between information
- Integrate Core Data with iCloud, so that when your data syncs between all your iOS devices
- Add a feature that employs Data Detectors - when you click on a phone number, date, or link, it can open the Phone, Calendar, or Browser app

Airplane Chat App

- A Wi-fi / Bluetooth chat app that allows you to message friends without needing an internet connection
- Uses the same technology as AirDrop
- Teaches the multi-peer connectivity framework
- It's great for camping and exploring outer space

Quora Replica

- Build an app that allows users to ask questions, write answers, and upvote the best answers
- This app teaches you to work with Firebase - the most popular mobile backend-as-a-service
- Firebase provides developers with an easy-to-use backend for your iOS app. (It's a place to store questions and answers in the cloud so that the app downloads them from the cloud.)

Foursquare Replica

- BlocSpot uses Apple Maps to discover new places and exciting adventure
- Search for places you want to visit, add them to a list, and get notifications when you get close to a place you've bookmarked
- This project teaches you to use Apple's Core Location API to get updates when the user changes location. It also teaches local notifications, which are like push notifications that don't require a backend

Swiftris 2

- Build Swiftris, Bloc's famous Tetris replica
- Add new features to Swiftris including high scores
- Integrate Game Center, and learn to use Game Center Challenges, Leaderboards, and Achievements
    - Game Center Challenges let you send friends your high score and challenge them to beat it. Practice writing trash talk
- Swiftris 2 features a revolutionary new game feature called "pause": when you switch to another app, the game pauses

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Publish your app to iTunes, and we'll help you promote it

Portfolio

- Create a unique Portfolio site that showcases your personality, projects, and writing
- Include images and links for your projects
- Build a blogging engine to explain your coding process and express your opinions
- Make your site stand out with a stunning design

Android Fundamentals

Build a Development Environment

- Learn how to use the command line
- Master version control and code management with Git and GitHub
- Configure your machine for Java development with Android Studio and the Java Development Kit (JDK)
- Learn the fundamentals of Java and Object-Oriented Programming concepts
- Learn Google Material Design - a set of design principles that were added in Android Lollipop - the most important version of Android since Honeycomb (3.0). One of the

major improvements was the release of Google Material Design - a set of interface principles for Web and Mobile apps

Java Fundamentals

- Master Java basics like variables, strings, arrays, conditional statements, and loops
- Learn about Object-Oriented Programming, classes, interfaces, inheritance and polymorphism
- Use collections, the abstract qualifier, and Java security protocols
- Comprehend multi-threading, overrides, and recursion

##Begin Your Android Journey

- Learn the building blocks of an Android application: Activities, Fragments, and Views
- Apply Google's Material Design to your applications
- Learn about UI, databases, and other standard application fundamentals

Your First Android App: Build an RSS Reader

- Create an interface with beautiful Android UI elements
- Learn to use Android's robust resource system
- Master the industry-standard MVC approach: Model-View-Controller
- Integrate Open Source libraries such as Universal Image Loader and jsoup
- Make your application pop with seamless animations
- Support tablet-sized devices with alternative layout

Technical Projects

You will complete at least two projects during the Project Phase. Projects are prescriptive yet open to interpretation and creativity. They are designed to challenge you while providing guidance to keep you on track. A project is complete when all its requirements are implemented and approved by your mentor.

BLOQUERY

- Build a replica of Quora
- Learn to incorporate Firebase - a popular mobile-backend as a service (MBaaS) to add server-side support to your Android application
- Leverage the Firebase API to store users, questions, and answers

BLOCLY EXPANDED

- In this project, you'll build upon the RSS Reader you created in Phase 1 to add additional features
- You'll make home screen widgets, background syncing services, and more

BLOCSPOT

- Integrate Google Maps into your app

- Use Yelp to locate points of interest and add them as BlocSpots
- Employ Google's Geofencing APIs to alert the user when they get close to their points of interest

BLOCPARTY
- Build a social photo app that uses the Instagram, Facebook, and Twitter API to aggregate images shared on each network into a single app
- Architect a multi-API login path
- Post photos to multiple networks at once

BLOCTALK
- Build a native SMS application from the ground up
- Query contacts and send both SMS and MMS messages

WHAT TO WEAR
- Use Android Wear APIs to build a wearable application
- Incorporate the OpenWeatherMaps API to provide forecast predictions and more
- Provide clothing suggestions to users based on up-to-day weather conditions

CAPSTONE
- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Publish your app to the Google Play Store, and we'll help you promote it

PORTFOLIO
- Create a unique Portfolio site that showcases your personality, projects, and writing
- Include images and links for your projects
- Build a blogging engine to explain your coding process and express your opinions
- Make your site stand out with a stunning design

UX/UI Design Fundamentals

UX AND UI FUNDAMENTALS
- Learn the importance of color theory, hierarchy, and balance
- Learn how to think like a designer
- Learn how to apply user-centered design
- Create user flows and design low-fidelity and high-fidelity wireframes
- Master the intangible skills of design, like presenting, selling, and critiquing

DESIGN TOOLS
As a professional designer you'll have many tools at your disposal. We'll teach you how to master the canonical tools that you'll use repeatedly as a professional.
- Photoshop - the prevalent design tool used in industry

- Illustrator - a vector-based graphics tool used for brand design, illustration, and wireframing
- Sketch - a responsive UI design tool
- Balsamiq - a popular wireframing tool
- InVision - a tool to create a clickable prototype of your web or mobile app, which is a great way to test usability and collect client feedback

CODING

- Designers should be generalists who can design and build elegant and useful interfaces. During the UX/UI Design part you'll learn the essential frontend markup languages
- HTML5 - HTML provides the scaffolding for web sites
- CSS3 - CSS allows you to style web sites and make them aesthetically pleasing
- Git, GitHub and GitHub Pages - you'll use Git for version control, GitHub to publish your code, and GitHub Pages to host live web sites
- Responsive Web Design - Understand the basics of responsive design and how to launch a website that naturally adapts to any device and any screen size

TESTING

- Create clickable prototypes of your designs
- Test your design choices with real data and user feedback, then incorporate feedback and revise your design
- Gather feedback from users and clients using InVision

MOBILE DESIGN

- Understand how web design translates to mobile design, and vice-versa
- Learn how to design apps for iOS and Android
- Design replicas of Spotify and Acorns for iPhone, and Yummly and WhatsAppfor Android

DESIGN PROJECTS

You will complete at least two projects during the Project Phase. Projects are prescriptive yet open to interpretation and creativity. They are designed to challenge you while providing guidance to keep you on track. A project is complete when all its requirements are implemented and approved by your mentor.

GrubHub for Food Trucks

- Research best practices for mobile payment apps
- Design user flows for discovering local food trucks and creating an order
- Create a mobile prototype to test and refine based on user feedback

Shopping

- Design an online shopping experience
- Create wireframes and mockups for consumers to browse and purchase products
- Build a responsive site that could be applied to Shopify's platform

Kickstarter for Nonprofits

- Research crowdfunding applications to create a competitive analysis
- Develop user personas of both funders and backers
- Design a complex UI that follows well-constructed user flows

Product Showcase

- Create a brand identity for a new product
- Design a product landing page to convey a value proposition and competitive differentiators
- Develop a responsive web site to deploy to GitHub Pages

Capstone

- Bring your product idea to life. Use your capstone project as an opportunity to hone your skills in a particular area of interest, or as a springboard to launch a new product
- Build your idea from scratch, while learning best practices for application scoping, design, and architecture
- Deploy your application

Portfolio

- Create a unique Portfolio site that showcases your personality, projects, and writing
- Include images and links for your projects
- Build case studies to discuss your design process and showcase your work
- Make your site stand out with a stunning design

JOB PLACEMENT ASSISTANCE

Career Services for Students of Track programs include:

- Mock Interviews
- Proactive introductions to recruiters and potential employers
- Careful monitoring of the job search process
- Troubleshoot and work on weak areas, including applications, phone-screens, or technical interviews
- Help with offer negotiations

Track students must meet a number of requirements to remain eligible for the Career Services that are offered:

- Complete all required assessments
- Complete all Programming Reinforcement Checkpoints
- Complete all Career Prep Checkpoints in the Career Prep project
- Apply to at least 10 jobs per week
- Keep track of all applications for review

- Provide weekly updates to the Outcomes Team and your mentor

STUDENT ACHIEVEMENT AND GRADUATION REQUIREMENTS

All programs require the student to complete 100% of the Foundation phase(s), plus all required projects. In addition to completing the Foundation phase(s) and required projects, the student must complete a specified number of elective projects in the Project phase.

Students of Bloc's Track programs are graded using a series of quizzes and in-person assessments, and students must pass each round of assessments before moving to the next phase of their program.

The project requirements for graduation are provided below:

Program	Number of Projects Required to Graduate
Android Mobile Development Fundamentals	1 Foundation Phase and 2 Android application Projects, approved by Mentor
UX/UI Design Fundamentals	1 Foundation Phase and 2 Design Projects, approved by Mentor
iOS Mobile Development Fundamentals	1 Foundation Phase and 2 iOS application Projects, approved by Mentor
Frontend Web Development Fundamentals	1 Foundation Phase and 2 Frontend development Projects, approved by Mentor
Rails Web Development Fundamentals	1 Foundation Phase and 2 Rails development Projects, approved by Mentor
Designer Track	3 Foundation Phases and 5 Design Projects, approved by Mentor
Part-Time Web Developer Track	3 Foundation Phases and 7 Web Development Projects, approved by Mentor

Software Developer Track	5 Foundation Phases and 7 Design Projects, approved by Mentor

SCHEDULE OF TOTAL CHARGES

Software Developer Track:
Estimated total costs for period of enrollment and program: $19,500

Web Developer Track:
Estimated total costs for period of enrollment and program: $8,800

Designer Track:
Estimated total costs for period of enrollment and program: $9,800

All Fundamental Programs:
Estimated total costs for period of enrollment and program: $5,000

FINANCIAL AID POLICIES

Bloc does not participate in federal or state financial aid programs and we do not provide institutional financing.

If a student receives a loan to pay for the educational program, the student will have the responsibility to repay the full amount of the loan plus interest, less the amount of any refund. Bloc does not offer institutional loans to its students. If the student receives federal student financial aid funds, the student is entitled to a refund of the money not paid from federal financial aid funds.

FACULTY

Each Bloc program is structured like an apprenticeship, wherein each student works closely with an experienced mentor to guide them in developing or designing software applications of increasing complexity. Bloc does not employ a traditional classroom format and does not have traditional faculty, but rather seasoned mentors who must have at least two years of real-world professional experience as a software developer or designer.

Instructor	**Course(s) Taught**	**Degree**	**Institution**	**Years of Technical Experience**
Aaron Uyehara	UX / UI	Bachelor of Art in International Cultural Studies Anthropology; Master in Science in Internet	Brigham Young University Hawaii Campus; Full Sail	9

		Marketing, SEO	University	
Adam Louis	Full Stack Track, Rails, Software Engineering Track	Self taught; Rails	NA; Bloc	5
Alex Spencer	Full Stack Track, Rails, Software Engineering Track	Self taught; Ruby/Rails Course	NA; Bloc	11
Alex Stophel	Full Stack Track, Rails, Software Engineering Track	Bachelor of Arts in History; Full Stack Web Development	King College; Bloc	2
Alissa Likavec	Frontend, Full Stack Track	Bachelor of Arts in Technical Writing and Mass Communications; Full Stack	University of South Florida; Bloc	5
Amber Aultman	UX / UI	Bachelor of Science, Psychology, Chemistry Minor; Fiber Design	University of Florida; Savannah College of Art and Design	6
Andrei Catinean	Android	Bachelor of Science in Computer Science	Universitatea Tehnică din Cluj-Napoca	4
Antonio Carella	Android, iOS	Master of Science in Computer Software Engineering	DePaul University	2
Benjamin Simmons	Frontend, Full Stack Track, Web-Developer Track, Rails	Bachelor of Arts in Philosophy and Religious Studies	University of Alabama	3
Bobbilee Hartman	Rails	Bachelor of Science in Business Administration, Marketing; Web Development	University of Arizona, Eller College of Management; The Starter League	3
Brandon Alexander	iOS	Bachelor in Science in Computer Science; Graduate work in Applied Computer Science	Missouri State University; Kennesaw State University	11
Brian Bugh	Full Stack	Self taught	NA	16

	Track, Rails, Frontend			
Brian Douglas	Rails	Bachelor of Science in Finance; Fullstack Knowledge, Rails, Javascript, jQuery	University of South Florida; Bloc	6
Brittany Martin	Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Marketing; Masters in Business Administration; Certificate in An Introduction to Interactive Programming in Python; Certificate in Ruby on Rails Web Development	University of Pittsburgh; Robert Morris University; Coursera; Bloc	6
Caron Stace	UX / UI	Bachelor of Arts in Graphic Design; Diploma in New Media; Frontend Bootcamp; Coding Bootcamp	Auckland Institute of Technology; Media Design School; Steer 2013; Makers Academy	16
Charlie Gaines	Full Stack Track, Rails, Software Engineering Track	Bachelor of Arts in Comparative Literature; Master of Science in Political Economy; Certificate Linux System Administration	University of California, Berkely; London School of Economics and Political Science; University of Illinois	8
Chris Baglieri	Rails	Bachelor of Engineering in Electrical Engineering; Master of Science, Bioinformatics	Villanova University; Northeastern University	19
Chris Gillis	Designer Track, UX / UI	Bachelor of Science in Entrepreneaurial Studies	University of Massachusetts, Amherst	16
Christopher Beck	Rails	Bachelor of Art in Visual Communications	University of North Carolina at Chapel Hill	17
Christopher Courtney	Designer Track, UX / UI	Bachelor of Arts in Communications and Ruby on Rails	University of Central Arkansas and Starter League	18
Clayton Liggitt	Rails, Software Engineering Track	Bachelor of Art in Political Science; Master of Art in Business Administration	University of California, Santa Barbara; Chapman University, The George L. Argyros School of Business	8

			and Economics	
Conrad Taylor	Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Mathematics and Computer Science; Certification in Machine Learning: Regression, and Machine Learning Foundations: A Case Study Approach, and Blasting-off-with-bootstrap	University of Illinois; Coursera and Code School	21
Cyle Ven Dawson	Frontend, Full Stack Track, Software Engineering Track	Bachelor of Science in Software Engineering	Iowa State University	6
Dalibor Ilijevski	Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Automatic Control	Faculty of Engineering University of Kragujevac	20
David Roberts	Full Stack Track, Rails	Bachelor of Science in Computer Science	University of Maryland College Park	9
Dennis Eusebio	UX / UI	Associate of Arts in Graphic Design; Bachelors of Fine Arts in Graphic Design	University of North Florida; University of Florida	12
Eliot Sykes	Rails	Bachelor of Engineering in Mechanical Engineering; Master of Science in Information Technology	University College London; Queen Mary, University of London	17
Ezekiel Binion	UX / UI	Bachelor of Fine Arts in Graphic Design; User Experience & Ruby/Ruby on Rails development	Columbia College Chicago; The Starter League	10
Falko Buttler	iOS	Master of Science in Computer Science	University of Bremen	14
Jason Dziak	UX / UI	Bachelor of Fine Art in Graphic Design and Computer Art	Bowling Green State University	18
Jason LaChapelle	Frontend	Self taught	NA	15
Jean-Denis Vauguet	Frontend, Full Stack Track, Rails	Bachelor of Applied Science in Mathematics and Computer Science	Université René Descartes (Paris V)	6
Jeff Lau	Designer Track, Frontend, Full	Bachelor of Arts Graphic Product Innovation	London College of Communications, University of Arts	6

	Stack Track, Software Engineering Track		London	
John O'Connor	Android, Frontend, Full Stack Track, Software Engineering Track	Bachelor of Science in Computer Science; Master of Business Administration	California State University - San Bernadino; UCI	12
John Sawers	Rails	Bachelor of Art in Computer Graphics / Choreography	Marlboro College	11
John Uke	Designer Track, Frontend, Full Stack Track, Software Engineering Track	Bachelor of Science in Business Administration, Entrepreneaurship/Enterpreneurial Studies; Engineering Bootcamp	University of San Diego, and Babson College; Dev Bootcamp	3
Jonathan Linowes	Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Computer Graphics and Film; Master of Science in Media Technology, Media Lab	Syracuse University; Massachusetts Institute of Technology	36
Jose Sanchez	Frontend, Full Stack Track, Web-Developer Track, Rails	Bachelor of Science in Computer Science; Full Stack Web Development	Florida International University; Bloc	8
Joseph Caudle	Frontend, Full Stack Track, Web-Developer Track, Rails	Bachelor of Arts in Philosphy, Theology, Medieval Studies	University of Notre Dame	4
Kevin McGillivray	Designer Track, Frontend, Full Stack Track, Software Engineering Track	Bachelor of Arts in Graphic Design and Music	St. Norbert College	4
Kinsey Durham	Web-Developer Track	Bachelor of Science in Advertising	University of Colorado at Boulder	3

Kurt Cunningham	Designer Track, Frontend, UX / UI	Bachelor of Arts in Journalism, History and English; Self-taught	University of Iowa	6
Levi Kennedy	Full Stack Track, Rails, Frontend	Bachelor of Arts in Commercial Music with an Emphasis in Music Technology	Belmont University	8
Luca Leone	UX / UI	Bachelor of Science in Economics	Bocconi University - Milano	5
Lukas Ingelheim	Web-Developer Track	Bachelor of Science in Economics & Finance	European Business School Oestrich-Winkel	3
Mark Carpenter	Designer Track, Frontend, Full Stack Track, iOS, Rails, Software Engineering Track	Bachelor of Business Administration in Finance/General	Grand Valley State University	22
Matt Born	Designer Track, Frontend	Bachelor of Fine Arts in Graphic Design	Illinois Institute of Art Schaumburg	10
Matt Hanlon	iOS	Bachelor of Arts in English	Vassar College	19
Matt Thompson	Frontend, Full Stack Track, Web-Developer Track, Rails	Associate's Degree in Network Specialist Associate's Degree in Web Development	Northeast Wisconsin Technical College	5
Matthew Knippen	iOS	Bachelor of Science in Computer Science	Illinois Institute of Technology	15
Matthew Maxwell	Designer Track, Frontend, Full Stack Track, Rails	Self taught	NA	11
Michael Pell	Designer Track, Frontend, Full Stack Track, Web-Developer Track, Rails,	Certificates in Getting and Cleaning Data, Exploratory Data Analysis, R Programming, and The Data Scientist's Toolbox	Coursera	4

	Software Engineering Track			
Michael Verde	Android, iOS	Bachelor of Applied Science in Engineering Physics	The University of British Columbia	12
Michał Kwiatkowski	Frontend, Full Stack Track, Rails	Master of Science in Computer Science	Politechnika Gdańska	5
Mike Dekker	UX / UI	Bachelor of Arts in Communication & Multimedia Design/Graphic Design; Bachelor of Art in Fine Arts; Bachelor of Art in Multimedia Design	Willem de Kooning Academie Rotterdam; Sint Joost Academie Breda; Grafisch Lyceum Rotterdam	18
Natalie Hatter	UX / UI	Bachelor of Arts in General Design, Graphic Design, Multimedia, Illustration	San Jose State University	21
Niki Brown	Designer Track, UX / UI	Bachelor of Fine Arts in Graphic Design	Iowa State University	9
Paul Jacobs	iOS	Bachelor of Science in Computer Science; Master of Art in Visual & Spatial Arts	University of California, Santa Barbara	13
Phillip Spitler	Frontend, Full Stack Track, Web-Developer Track, Rails, Software Engineering Track	Self taught	NA	17
Rajeev Singh	Web-Developer Track	Self taught	NA	13
Richard Newman	Frontend, Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Computer Science; Masters of Business Administration in Business	Boise State University; University of Washington, Michael G. Foster School of Business	20
Ricky Panzer	iOS, Rails	Bachelor of Science in Computer Science and Business	Cornell University	5
Ryan Balfanz	Designer Track,	Bachelor of Science in Physics, Mathematics; Certificates for Try	Illinois State University; Code	7

	Frontend, Full Stack Track	Git, Shaping-up-with-angular-js and JavaScript Road Trip Part 1	School	
Sarah Harrison	UX / UI	Bachelor of Fine Arts in Interface Design and Graphic Design	University of Idaho	14
Shannon Bertucci	Frontend, Full Stack Track, Rails, Software Engineering Track	Bachelor of Science in Computer Science	University of Maryland College Park	9
Sharon Torres	Designer Track, UX / UI	Bachelor of Fine Arts	Fullerton College	18
Steven Schauer	iOS, Rails	Self taught	NA	15
Terry Million	Designer Track, UX / UI	Bachelor of Science in Computer Graphics Technology, Visual Communications & Interactive Multimedia	Indiana University - Purdue University of Indianapolis	9
Timothy Barnes	Web-Developer Track, Rails	Associate of Applied Science in Architectural Drafting and Design & C.A.D. Technology; Bachelor of Arts in Speech Communication and Rhetoric; Web Development	Lincoln Technical Institute-Indianapolis; Arkansas State University; Bloc	10
Wilson Rector	Web-Developer Track	Self taught	NA	9
Xander Miller	Full Stack Track, Rails, Software Engineering Track	Bachelor of Arts in Cognitive Science; Certifications in Mastering-github and Rails 4	Carleton University; Code School	17
Zach Zimbler	Frontend	Bachelor of Science in Information Sciences & Technology + Engineer Design Minor	Pennsylvania State University	4

STUDENT GRIEVANCE POLICY

Bloc encourages students to bring all complaints or grievances about academically related situations to its attention. Many questions or concerns that students may have can be resolved simply through discussion.

A student may present a grievance through the following complaint and dispute resolution procedures. Bloc will investigate all complaints or grievances fully and promptly.

A grievance is defined as a student's written expression of dissatisfaction concerning conditions of enrollment or treatment by mentors, other students, or staff. Grievances may include misapplication of Bloc's policies, rules, regulations, and procedures, or unfair treatment.

STEP 1
A student should first bring the grievance to the attention of their mentor or contact help@bloc.io.

STEP 2
Should the student's grievance not be resolved to the student's satisfaction after completing step 1, the student should next bring the grievance to the attention of the Chief Executive Officer.

STEP 3
At any time, the student may contact the BPPE with concerns or complaints:

Bureau for Private Postsecondary Education
P.O. Box 980818
West Sacramento, CA 95798-0818
Phone: 916-431-6959
Fax: 916-2631897
Website: www.bppe.ca.gov

STUDENT SERVICES

Bloc students have access to the following services:

- Curriculum and curated reference material on the Program Roadmap
- Career Services
- Online Student Forums (Slack, Facebook, LinkedIn)
- Student Advisers
- Support and Issue Resolution

CANCELLATION AND REFUND POLICIES

STUDENT'S RIGHT TO CANCEL

1. You have the right to cancel your agreement for a program of instruction, without any penalty or obligations, and receive a full refund, before the first lesson and materials are received, or within one week of your Program Start Date, whichever is later. If Bloc provided the first lesson and materials before an effective cancellation notice was received, Bloc shall make a refund within 45 days after you return the materials. If no materials are required, then the right to cancel shall be through the seventh calendar day after enrollment. After the end of the cancellation period, you also have the right to stop school at any time; and you have the right to receive a pro rata refund if you have completed 60 percent or less of the scheduled hours in the current payment period in your program through the last day of attendance.

2. Cancellation may occur when the student provides a written notice of cancellation at the following address: Student Support and Operations, 110 Sutter, 10th Floor, San Francisco, CA 94104 –OR— help@bloc.io. This can be done by mail, email or by hand delivery.
3. The written notice of cancellation, if sent by mail, is effective when deposited in the mail properly addressed with proper postage.
4. The written notice of cancellation need not take any particular form and, however expressed, it is effective if it shows that the student no longer wishes to be bound by the Enrollment Agreement.
5. If the Enrollment Agreement is cancelled the school will refund the student any pro-rated money he/she paid, less a registration or administration fee not to exceed $250.00, and less any deduction for equipment not returned in good condition, within 45days after the notice of cancellation is received.

WITHDRAWAL FROM THE PROGRAM

You may withdraw from the school at any time after the cancellation period (described above) and receive a pro rata refund if you have completed 60 percent or less of the scheduled days in the current payment period in your program through the last day of attendance. The refund will be less a registration or administration fee not to exceed $250.00, and less any deduction for equipment not returned in good condition, within 45 days of withdrawal. If the student has completed more than 60% of the period of attendance for which the student was charged, the tuition is considered earned and the student will receive no refund. For the purpose of determining a refund under this section, a student shall be deemed to have withdrawn from a program of instruction when any of the following occurs:

- The student notifies the institution in writing of the student's withdrawal or as the student's actual last date logged in or last date of an academically-related activity, whichever is later.
- The institution terminates the student's enrollment for failure to maintain satisfactory progress; failure to abide by the rules and regulations of the institution; absences in excess of maximum set forth by the institution; and/or failure to meet financial obligations to the School.
- The student has failed to attend class for three (3) consecutive weeks.
- The student fails to return from a leave of absence.

For the purpose of determining the amount of the refund, the date of the student's withdrawal shall be deemed the last date of recorded attendance. The amount owed equals the daily charge for the program (total institutional charge, minus non-refundable fees, divided by the number of days in the program), multiplied by the number of days scheduled to attend, prior to withdrawal. For the purpose of determining when the refund must be paid, the student shall be deemed to have withdrawn at the end of three (3) consecutive weeks without attendance; however, the refund amount will be calculated using the student's actual last date logged in or last date of an academically-related activity, whichever is later. If the student has completed more than 60% of the period of attendance for which the student was charged, the tuition is considered earned and the student will receive no refund.

STUDENT TUITION RECOVERY FUND (STRF)

The State of California created the Student Tuition Recovery Fund (STRF) to relieve or mitigate economic losses suffered by students in educational programs who are California residents, or are enrolled in a residency program attending certain schools regulated by the Bureau for Private Postsecondary Education.

You may be eligible for STRF if you are a California resident or are enrolled in a residency program, prepaid tuition, paid STRF assessment, and suffered an economic loss as a result of any of the following:

1. The school closed before the course of instruction was completed.
2. The school's failure to pay refunds or charges on behalf of a student to a third party for license fees or any other purpose, or to provide equipment or materials for which a charge was collected within 180 days before the closure of the school.
3. The school's failure to pay or reimburse loan proceeds under a federally guaranteed student loan program as required by law or to pay or reimburse proceeds received by the school prior to closure in excess of tuition and other costs.
4. There was a material failure to comply with the Act or the Division within 30 days before the school closed or, if the material failure began earlier than 30 days prior to closure, the period determined by the Bureau.
5. An inability after diligent efforts to prosecute, prove, and collect on a judgment against the institution for a violation of the Act.

However, no claim can be paid to any student without a social security number or a taxpayer identification number.

You must pay the state-imposed assessment for the Student Tuition Recovery Fund (STRF) if all of the following apply to you:

1. You are a student in an educational program, who is a California resident, or are enrolled in a residency program, and prepay all or part of your tuition either by cash, guaranteed student loans, or personal loans, and
2. Your total charges are not paid by any third-party payer such as an employer, government program, or other payer unless you have a separate agreement to repay the third party.

You are not eligible for protection from the STRF, and you are not required to pay the STRF assessment, if either of the following applies:

1. You are not a California resident, or are not enrolled in a residency program, or
2. Your total charges are paid by a third party, such as an employer, government program, or other payer, and you have no separate agreement to repay the third party.

A student seeking reimbursement under the Student Tuition Recovery Fund must file a written application on the Bureau of Private Postsecondary Education's Student Tuition Recovery Fund Application Form, available at www.bppe.ca.gov, signed under penalty of perjury that the form and all attachments are true and correct. Students must complete and file the STRF application form and all supporting documents with the Bureau within two years of receiving a closure

notice explaining the student's rights under STRF, or within a maximum of four years if the student received no closure notice.

RECORDKEEPING

Bloc maintains a file for each student who enrolls in the institution whether or not the student completes the educational service. Student records are maintained for a minimum of five years from the student's date of completion or withdrawal, with progress and performance data, and completion certificate, including a student transcript, maintained indefinitely. Bloc maintains and retains all records required by The California Private Postsecondary Education Act of 2009 ("the Act"). Student records required by the Act are maintained in the state of California, and stored in digital software in a manner secure from damage or loss. Bloc will take reasonable steps to protect the privacy of personal information contained in student records.

All student records will be made immediately available by the institution for inspection and copying during normal business hours by the Bureau of Private Postsecondary Education and any entity authorized to conduct investigations. If Bloc closes, it will arrange for the storage and safekeeping in California of all records required to be maintained by the Act for as long as those records must be maintained.

Student may request to review their student records, or a copy of their completion certificate of transcript by contacting help@bloc.io.

UNANSWERED QUESTIONS

Any questions a student may have regarding this catalog that have not been satisfactorily answered by the institution may be directed to the Bureau for Private Postsecondary Education at:

Address: 2535 Capitol Oaks Drive, Suite 400, Sacramento, CA 95833
P.O. Box 980818, West Sacramento, CA 95798-0818
Web site Address: www.bppe.ca.gov
Telephone Number: (888) 370-7589
Fax Number: (916) 263-1897

COMPLAINT PROCESS

A student or any member of the public may file a complaint about this institution with the Bureau for Private Postsecondary Education by calling the Toll-free telephone #: (888) 370-7589 or by completing a complaint form, which can be obtained on the bureau's internet Web site Web site Address: www.bppe.ca.gov.