

Timeline of History App

Authors:

Kacper Akdag-Ochnik, Anycode Student Club, Technical University of Koszalin

Tomasz Góralski, Anycode Student Club, Technical University of Koszalin

Tomasz Lech, Anycode Student Club, Technical University of Koszalin

Krzysztof Niemiec, Anycode Student Club, Technical University of Koszalin

Code Repositories:

<https://github.com/anycode-pk/bitnbuild-back>

<https://github.com/anycode-pk/bitnbuildfront>

Live Versions:

<https://bitnbuildfront.web.app>

<https://shirotsuma.pythonanywhere.com>

Problem Statement

The challenge we address is the lack of engaging and comprehensive tools for learning history in a chronological context. Existing resources may lack interactivity and fail to provide an immersive experience.

History is an essential component of education, fostering a deep understanding of societal development and cultural evolution. However, traditional learning methods may not fully capture the attention of users, leading to disinterest and suboptimal retention of historical facts.

With the advancement of technology, there is an untapped potential to create an application that not only provides a detailed historical timeline but also incorporates interactive modules and games. This approach has the potential to bridge the gap between traditional learning methods and the preferences of modern users.

Solution Description

Our team has created a dynamic historical timeline educational application with the following key features and functionalities:

Thematic Modules and Interactive Timeline:

The application organizes historical facts into thematic modules, providing a structured overview of diverse topics. An interactive timeline presents events chronologically, ensuring a cohesive learning experience.

Engaging Presentation and Mini-Games:

Utilizing visually captivating graphics, animations, and multimedia elements, the application offers an engaging presentation of historical events. Interactive mini-games related to historical contexts enhance user involvement, making the learning process both informative and enjoyable.

Technology stack

Front end

- Flutter (Dart)
 - Extensive library of components and packages
 - Cross-platform
 - A modern and elegant solution
- Firebase (hosting)
 - Super fast deployment
 - Free for startups
 - Ease of use

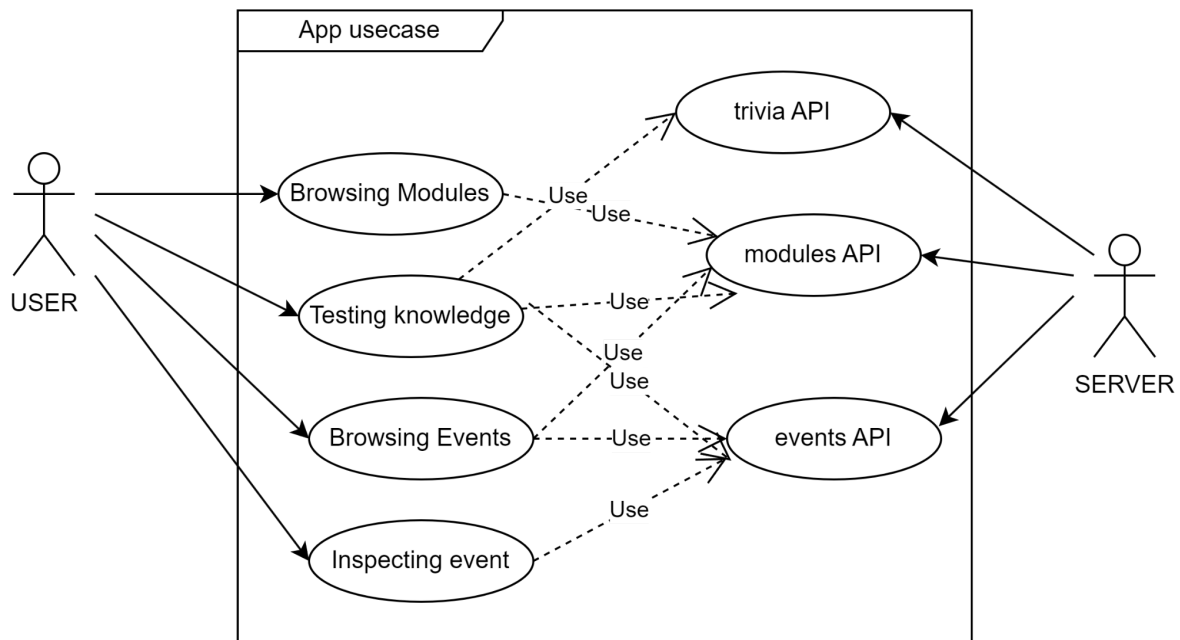
Back end

- Flask (python)
 - Lightning fast prototyping and development
 - Ultra light syntax
- SQLite database
 - Fast
 - Lightweight
 - SQL
 - Local file
 - Simple types
- PythonAnywhere (hosting)
 - Simple flask deploy
 - Free tier
 - Good response time

Architecture and Design:

The project is separated into two sub projects covering the backend and frontend respectively. The backend is written in Python and conforms to the RESTful API. Endpoints pointing to databases implement CRUD through POST, GET, PUT and DELETE. Data such as learning modules, events in timeline and image urls are queried through specific endpoints.

The frontend section is written in Flutter and hosted on Firebase. It handles querying and displaying data from API endpoints through specified URLs. The application is written mainly for mobile device browsers, but fully supports normal browsers and can potentially be deployed as a native mobile app due to the flexibility Flutter offers to the developer.



App use case diagram

Codebase Overview:

Frontend overview

Source code sits in the `/lib` folder as per Flutter development guidelines.

main.dart: the entry point of the app.

/lib/module: code related to the module view (main screen).

/lib/timeline: implements the timeline shown for each module after selecting it.

/lib/event: implements the event view that can be seen after clicking a card in the timeline view.

lib/minigames: all of the code regarding the educational minigames. Defines all game screens, code for randomizing the games' data, as well as reusable tiles seen in all of them.

Backend overview

run.py: entry point of application. It launches the backend from packages.

requirements.txt: This file lists all Python dependencies that the project needs.

packages/backend.py: This is a Python module that contains the logic of application. It's part of a package, which is indicated by the presence of the `__init__.py` file.

databases/app.sql: This file contains SQL code related to the database initialization by application.

databases/app.db: This file contains the whole database used by the app as a local file.

templates/index.html: This is an HTML file that serves as a template for a web page in application.

static/images/: This directory contains static files (like images) that are hosted on github.

.gitignore: This file tells Git which files or directories to ignore in the project.

Installation and setup:

Frontend

1. Install dart
2. Install flutter sdk
3. Clone github repository: <https://github.com/anycode-pk/bitnbuildfront>
4. Run `flutter pub get`
5. Run `flutter run`

Backend

1. Install python
2. Clone github repository: <https://github.com/anycode-pk/bitnbuild-back>
3. Run `pip install -r requirements.txt`
4. Run `flask -app run run in project repository`

Usage Guide:

Depending if the app is hosted locally or if used through a hosted [app](#) we get access to an app that allows users to learn and play around straight away. On default the user connects to the hosted [database](#) and gets data straight from it. It's displayed straight away and doesn't require from the user anything other than internet connection.


Using the app is rather straightforward - the first screen greets the user with a selection of modules to learn from. After clicking one, the user sees a timeline showing all of the events contained in the module. Clicking any event shows the user additional details.

The timeline view also features a "play" button that generates a set of 10 random minigames to test the user's knowledge in an interactive way.


×

DEBUG


Match the pairs




1790



1758



1782



1773

2

2


1

1

Submit


Modules

DEBUG




Polish Rulers

Polish kings



Polish Uprisings

Polish uprisings



Presidents of USA

←

Timeline

DEBUG

1830

November Uprising

Also known as the Polish–Russian War of 1830–31 or t...

1863

January Uprising

Just 32 years after the failed November Uprising, another ...

1918

Wielkopolska Uprising

World War One shook a lot of things up in Europe, seeing t...

1919

Łódź Revolution

Bet you've never heard about this before?! You may be as ...

1919

Silesian Uprisings

After Wielkopolska, another post-WWI uprising followed i...

1943

Warsaw Ghetto Uprising

+


←

Details

DEBUG

Łódź Revolution

1919



Bet you've never heard about this before?! You may be as surprised as we were when we first learned of Łódź's own attempt at revolution. But that's what you get when you mix a working class culture with a hatred for Imperial Russia. So it proved in 1905, when the people of Łódź rose in rebellion against their Tsarist rulers. Russia's disastrous military campaign against Japan in the Russo-Japanese War (won handily by Japan) had far-reaching consequences, battering an already fragile