<BilliardTech>

<Project Name>
Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
<22/07/2023>	<1.0>	<Create Software Architecture Document>	<Le Tran Hoang Phuc>

Table of Contents

Software Architecture Document

1. Introduction

The introduction of the Software Architecture Document provides an overview of the entire Software Architecture Document. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the Software Architecture Document.

- **Scope**

  The Software Architecture Document's scope encompasses outlining the software system's extensive-level technology and design decisions.The document's primary goal is to offer a clear understanding of the system's structural components, including computational elements and their interactions facilitated by connectors. By defining the document's purpose, objectives, and overall overview, it ensures a comprehensive and well-structured representation of the software's architectural aspects.

- **References**

  Vision document

  Use-case model

  Use-case specification

- **Overview**

  The software architecture includes:

  1. Introduction
  2. Architectural goal and constraints
  3. Use-case model
  4. Logical overview

2. Architectural Goals and Constraints

General Requirements:

- UI: use figma to design simple UI
- System: Django
- Database: MySQL

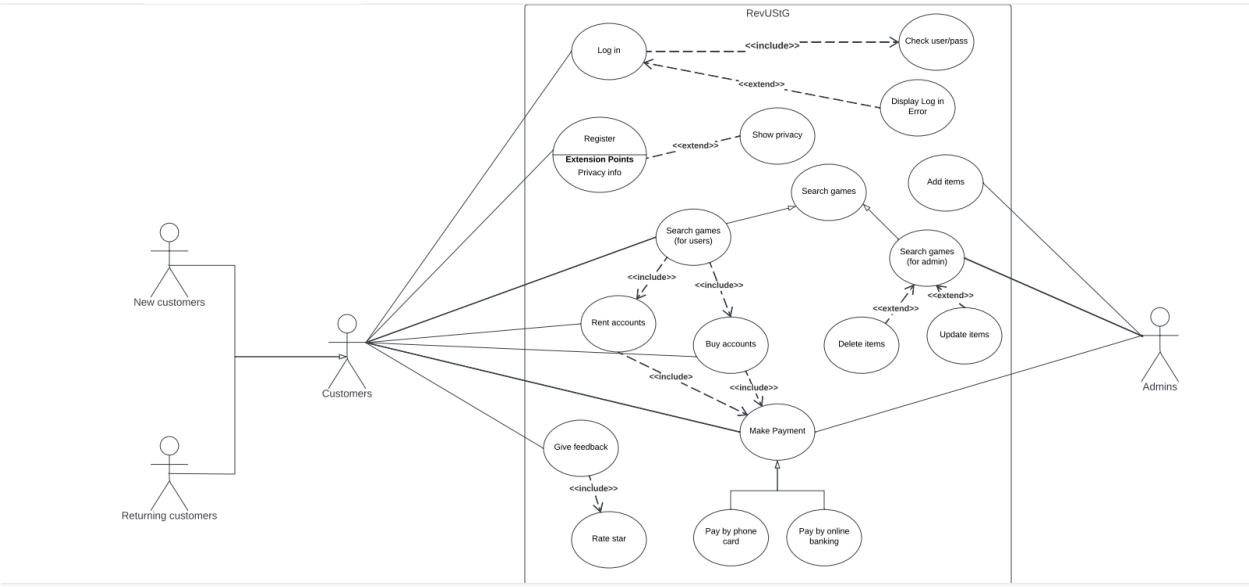
Applicable standards:

- User Testing: Regularly test the application with real users to gather feedback and improve the UI/UX.
- Performance Optimization: Optimize the application's performance to reduce loading times and resource usage. Enable users to have better experiences while buying and renting accounts.
- Consistency: Maintain a consistent design throughout the application for better user navigation.

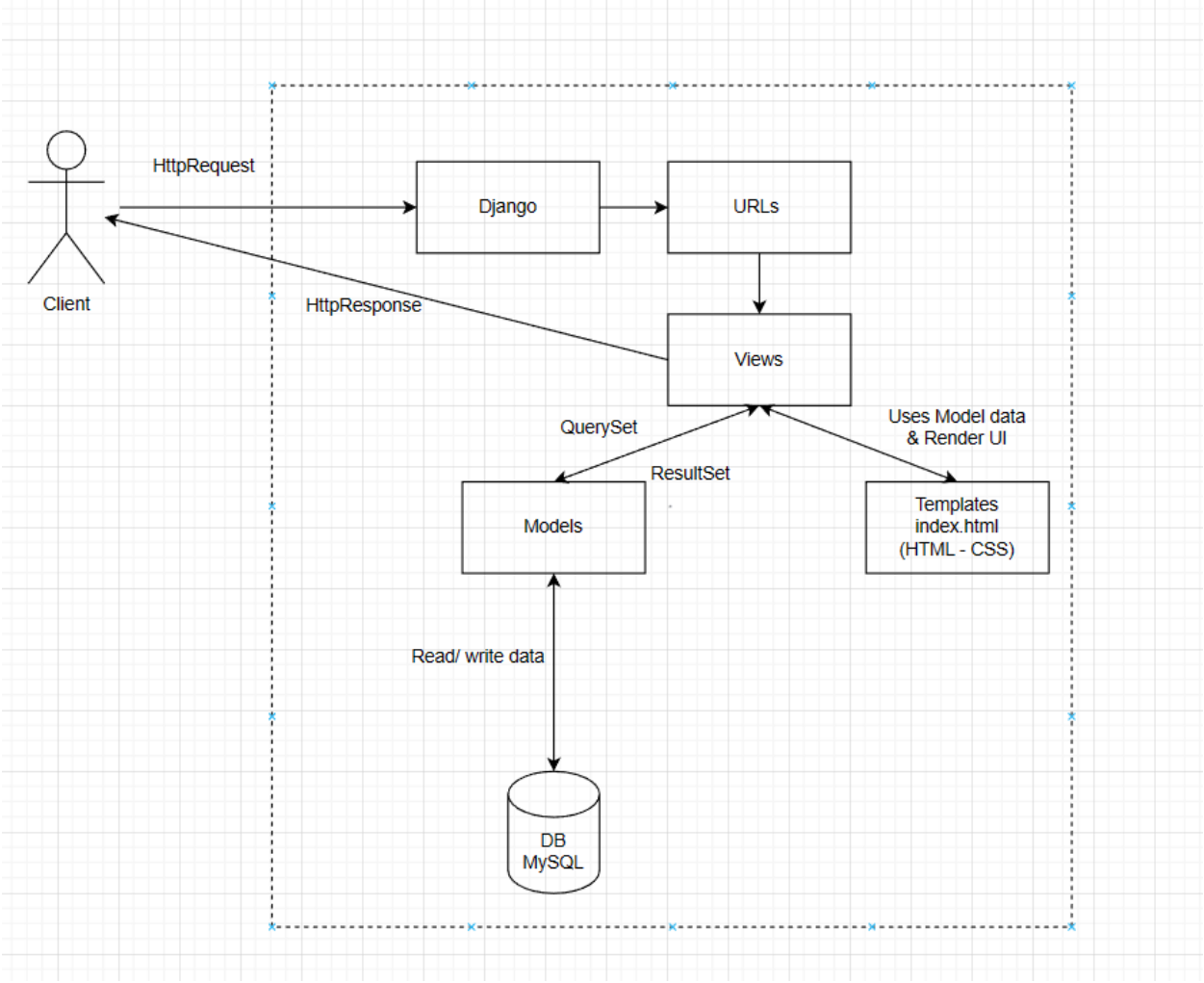
Security:

- Security: Prioritize security measures to protect user data and privacy.

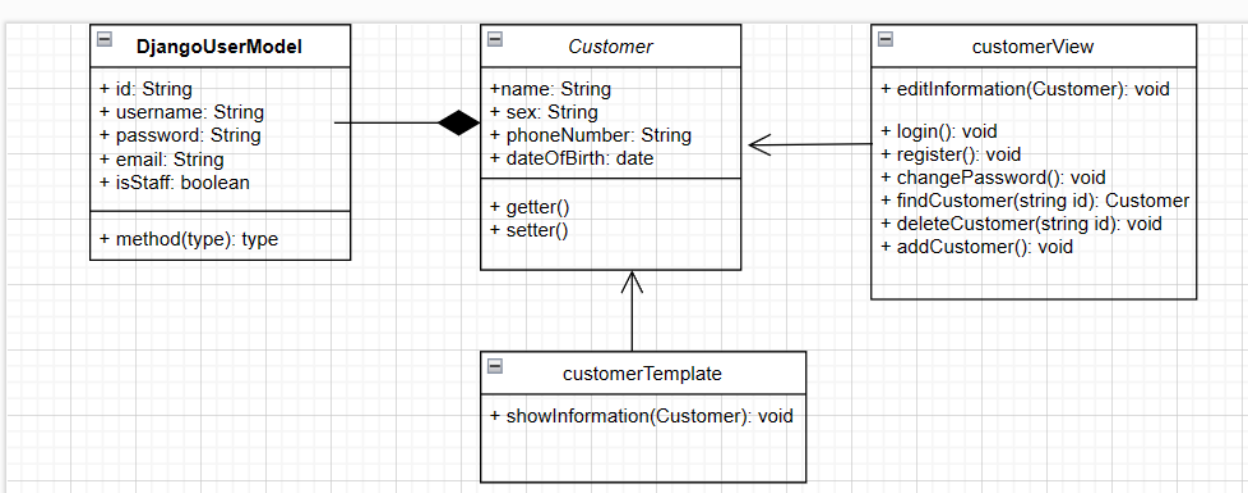
3. Use-Case Model



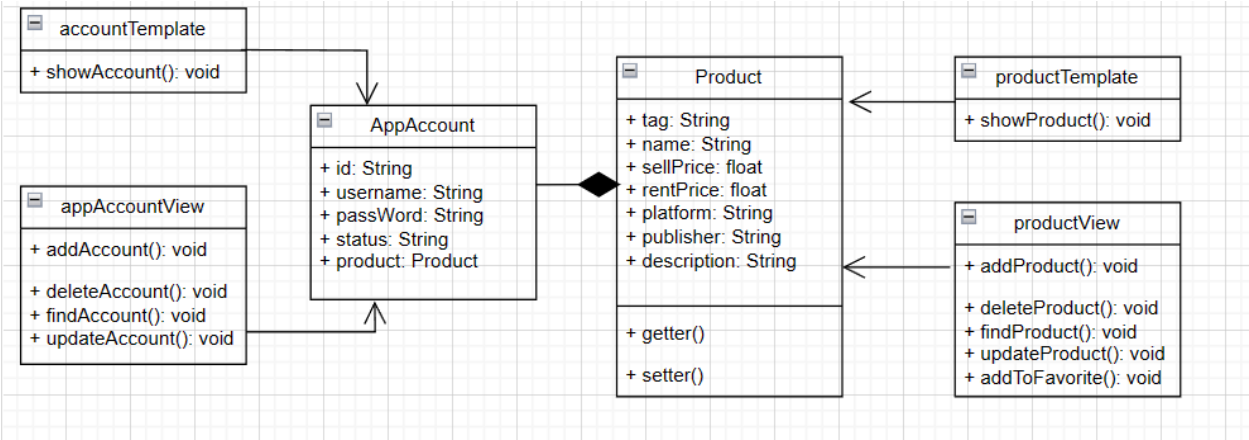
4. Logical View



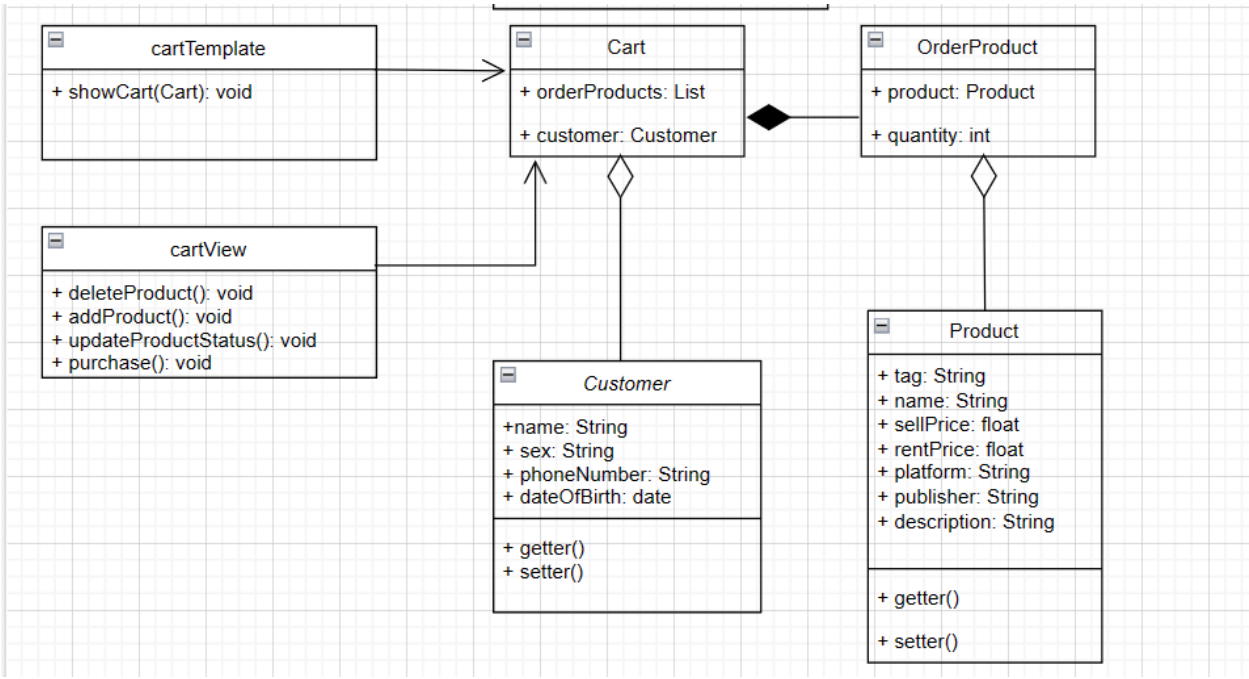
4.1 Component: Customer



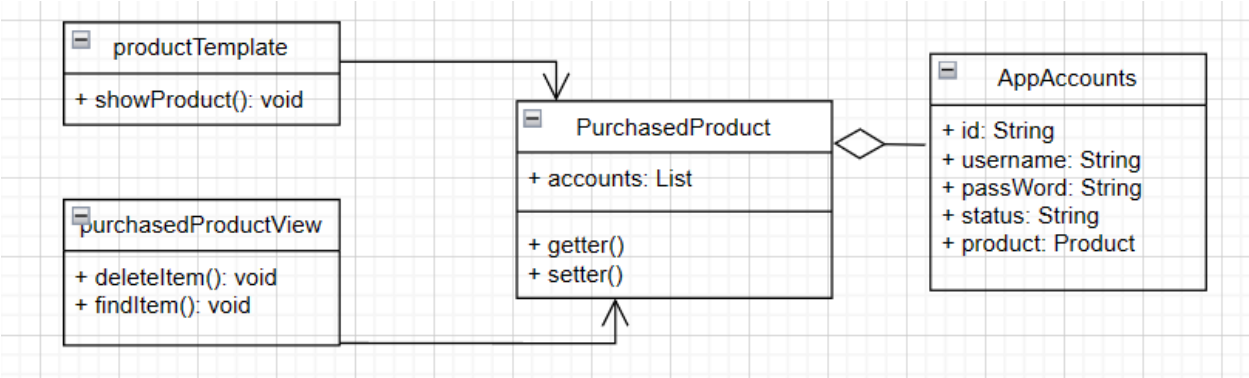
4.2 Component: Product



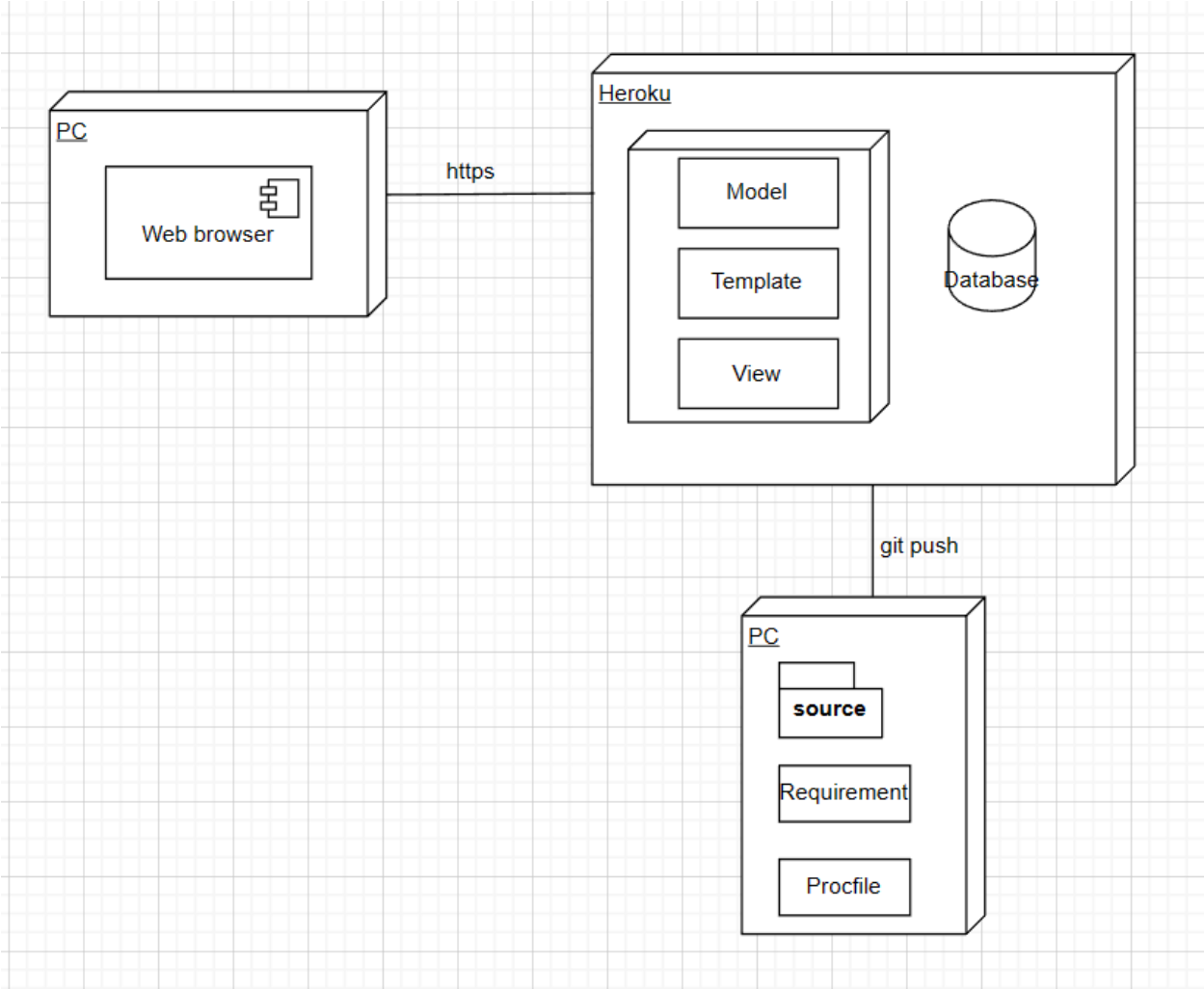
4.3 Component: Cart



4.4 Component: Purchased



5. Deployment



6. Implementation View

RevUStG: The main folder including all sub-folders of the project

RevUStG: the folder created by django including files that configure the Web

settinggs.py: configure the path of static file, apps created, database path, templates path, ...

urls.py: URL configuration for project

accounts: the folder created to handle all parts of the web

models.py: the file where models are defined

admin.py: use to register the models to django admin site

form.py: use to store all the class creating forms using on the Web

urls.py: configure the URL using for each page

views.py: retrieves the data from the models in views and renders that data with a t template, processes the data with our logic and gives that data to the template.

static: folder store all the static files using for the web, including css, js, images

templates: including all the html files that will be render through views.py

manage.py: automatically built file created by django, establishes your project's package on sys.path, fixes the DJANGO_SETTING_MODULE environment variable to point to your project's setting.py file.