TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

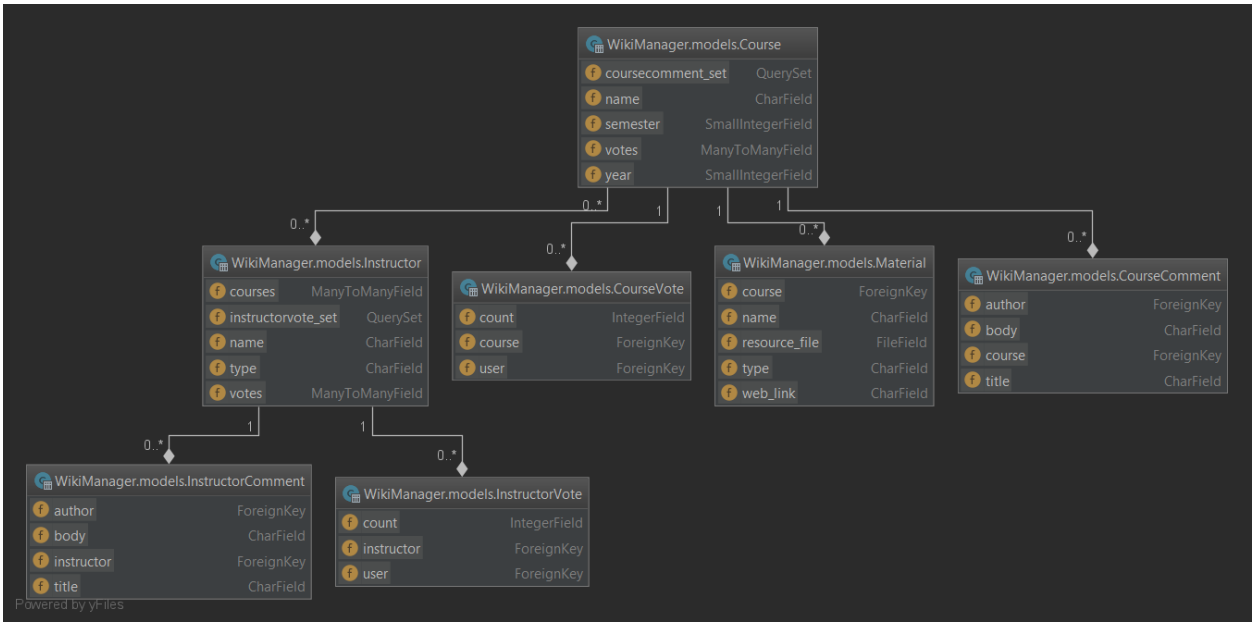
# UTCN-Wiki

Student: Marasescu-Duran Antonio

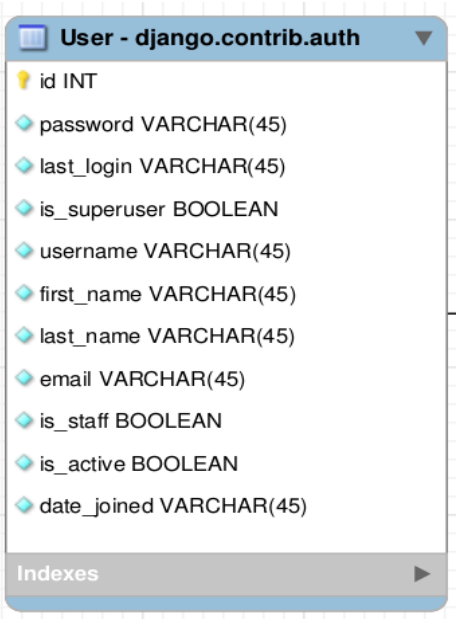
Group: 30431

# 1 Domain Model Diagram

In Django the Database tables are described through classes in the *models.py* of the specific Django App. This in turn allows the generation of the **Django Model-Dependency-Diagram**.




All of the fields called "**user**" are a *foreign key* to the **Django built-in User Model** described below:

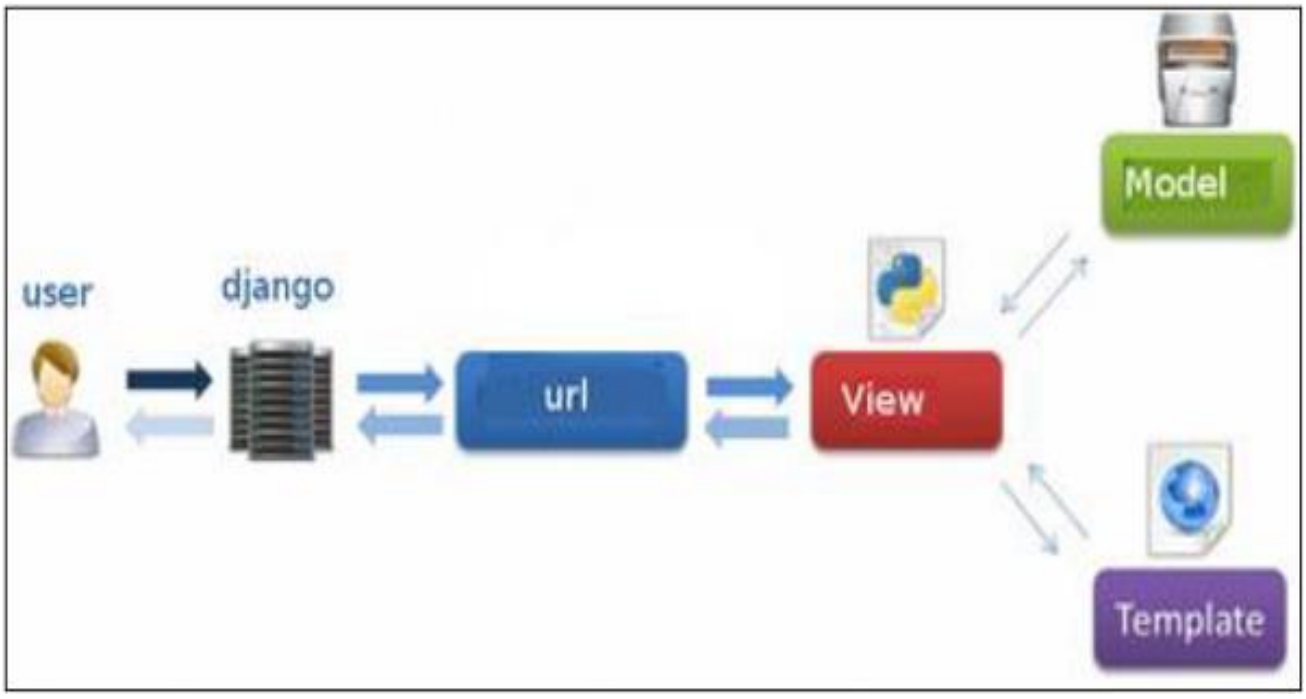


# 2 Conceptual Architecture

Our architecture will be separated into a **Django Backend** (which will load a single *HTML template*) and a **React Frontend** which will manage said *template*. The reason why **Django** was chosen was due to its quick development curve for small projects. Moreover, the **React** part of our architecture was chosen for the reusability part of the implemented components and the ability to better organize and manage said components.

## 2.1 Django MVT Pattern

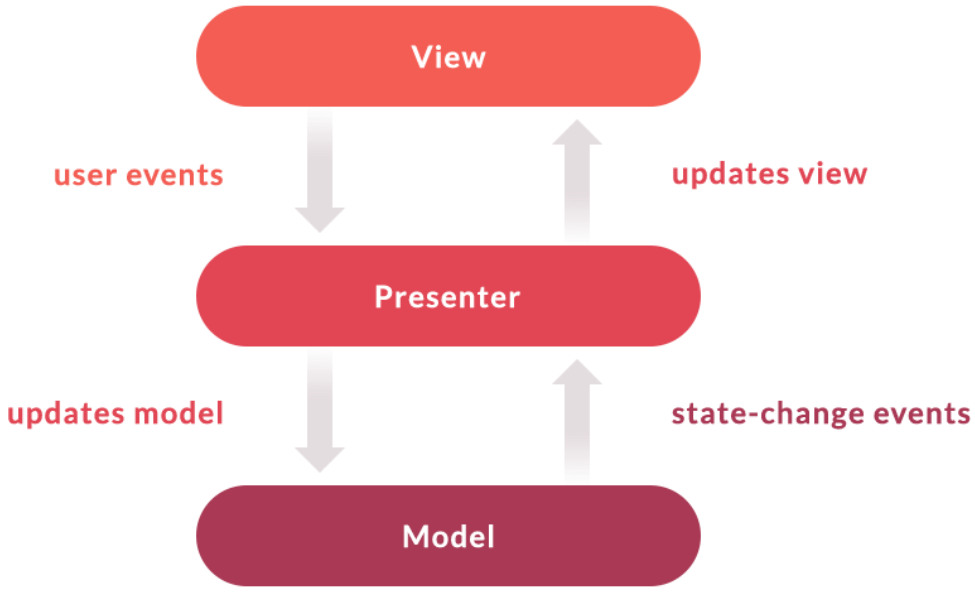
The **Model-View-Template** (MVT) is slightly different from MVC. In fact, the main difference between the two patterns is that **Django** itself takes care of the ***Controller*** part (controls the interactions between the Model and View), leaving us with the **Template**. The **Template** is a HTML file mixed with Django Template Language (DTL).




Django MVT Pattern Interaction

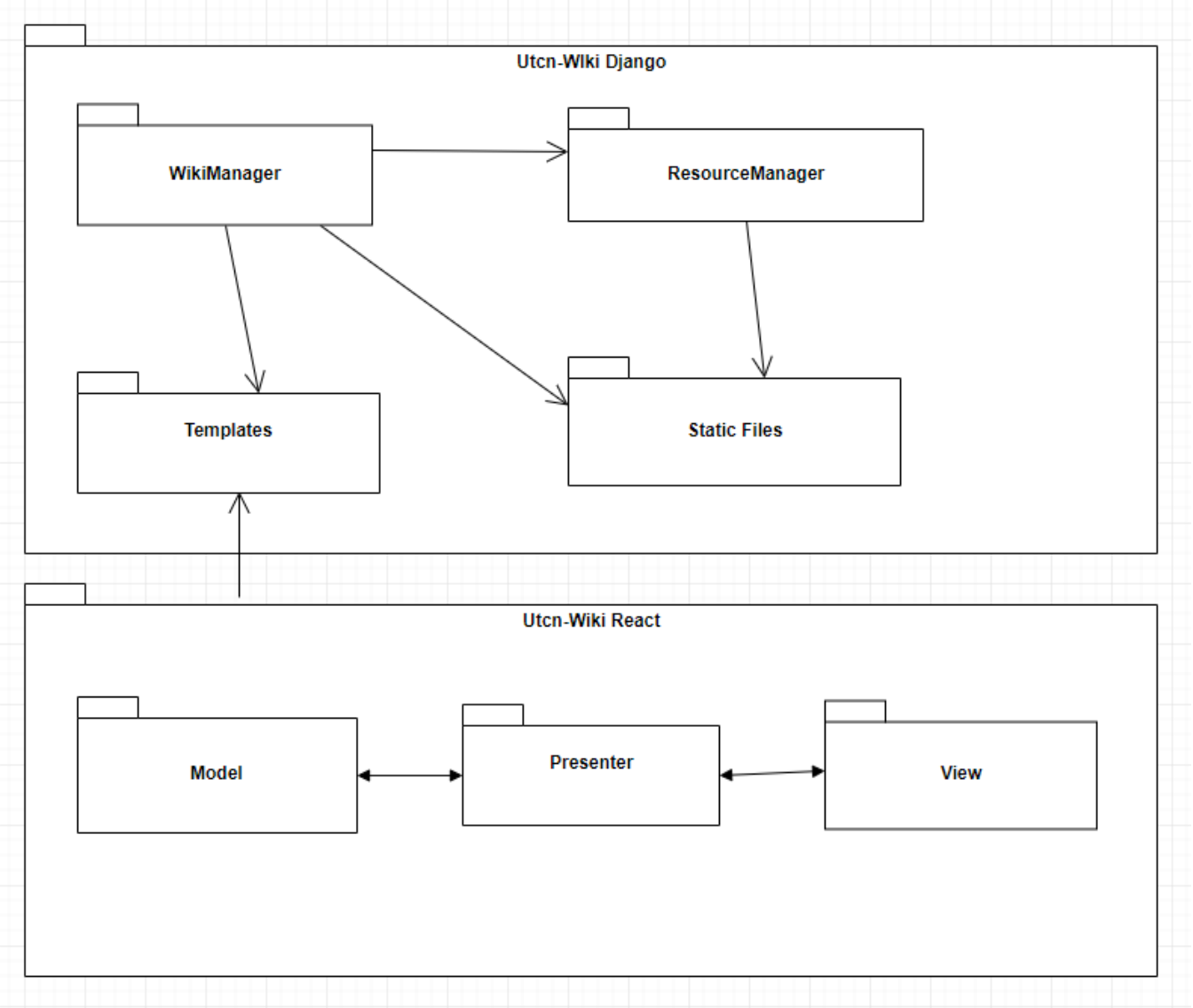
## 2.2 React MVP Pattern

React will implement the Model-View-Presenter architectural pattern. This is derived from Model-View-Controller pattern, the main difference is that, while in MVC the model fires updates in the view, in MVP it only relies on the presenter. We have chosen this pattern due to the fact that it breaks the connection the Model had with View in the original MVC, therefore creating only one class that handles everything related to the presentation of the View. This allows for a more easier integration into the React Framework and a more organized system.




[React MVP Presentation]

# 3 Package Design




Django uses the concept of reusable apps, this are web applications that can act with a high degree of independence. In our system such web apps are **WikiManager** and **ResourceManager**.

# 4 Components and Deployment