In [151]:

```
from sklearn.preprocessing import StandardScaler
Scaler = StandardScaler()
x_reduced_trainStand= Scaler.fit_transform(reduced_x_train)
x_testStand= Scaler.fit_transform(x_test)
x_valStand=Scaler.fit_transform(x_val)
```

In [158]:

```
from keras.models import Sequential
from keras.layers import Dense
```

In [162]:

```
model = Sequential([
    Dense(32, activation='relu', input_shape=(15,)),
    Dense(32, activation='relu'),
    Dense(1, activation='sigmoid'),
])
```

In [163]:

```
model.compile(optimizer='sgd',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

In [164]:

```
hist = model.fit(x_reduced_trainStand, reduced_y_train,
          batch_size=32, epochs=100,
          validation_data=(x_valStand, y_val))
```

```
- accuracy: 0.9619 - val_loss: 0.1053 - val_accuracy: 0.9597
Epoch 33/100
270/270 [==============================] - 1s 4ms/step - loss: 0.1011
- accuracy: 0.9618 - val_loss: 0.1053 - val_accuracy: 0.9611
Epoch 34/100
270/270 [==============================] - 1s 4ms/step - loss: 0.1004
- accuracy: 0.9629 - val_loss: 0.1065 - val_accuracy: 0.9600
Epoch 35/100
270/270 [==============================] - 1s 5ms/step - loss: 0.1001
- accuracy: 0.9623 - val_loss: 0.1039 - val_accuracy: 0.9624
Epoch 36/100
270/270 [==============================] - 1s 3ms/step - loss: 0.0997
- accuracy: 0.9629 - val_loss: 0.1082 - val_accuracy: 0.9584
Epoch 37/100
270/270 [==============================] - 1s 3ms/step - loss: 0.0992
- accuracy: 0.9629 - val_loss: 0.1027 - val_accuracy: 0.9616
Epoch 38/100
270/270 [==============================] - 1s 3ms/step - loss: 0.0987
- accuracy: 0.9622 - val_loss: 0.1078 - val_accuracy: 0.9586
```

In [165]:

```
model.evaluate(x_valStand, y_val)[1]
```

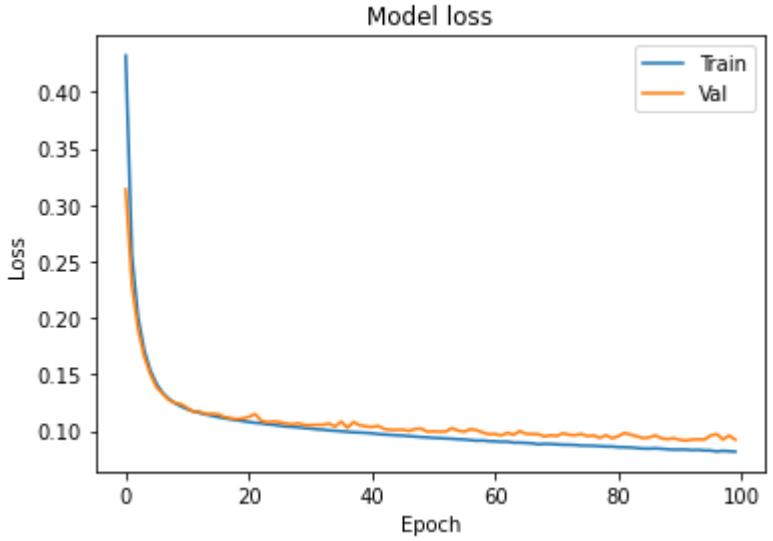
```
116/116 [==============================] - 0s 2ms/step - loss: 0.0923
- accuracy: 0.9619
```

Out[165]:

```
0.9618712663650513
```

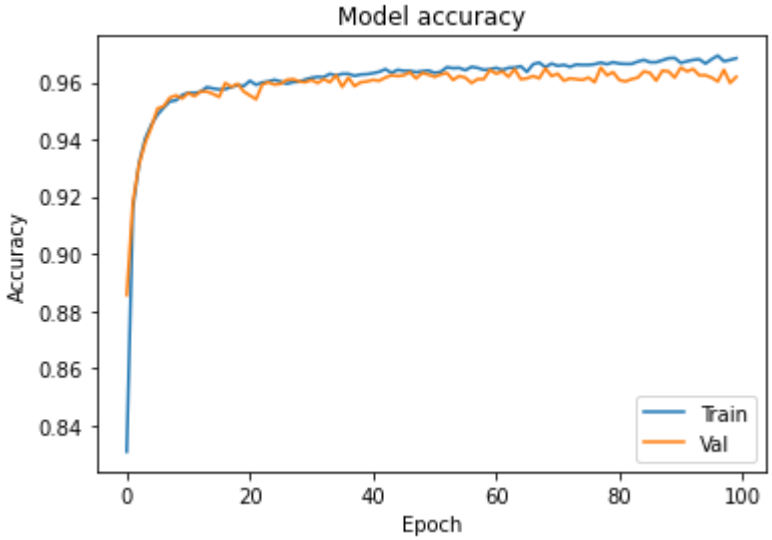
In [166]:

```
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.show()
```



In [169]:

```
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='lower right')
plt.show()
```



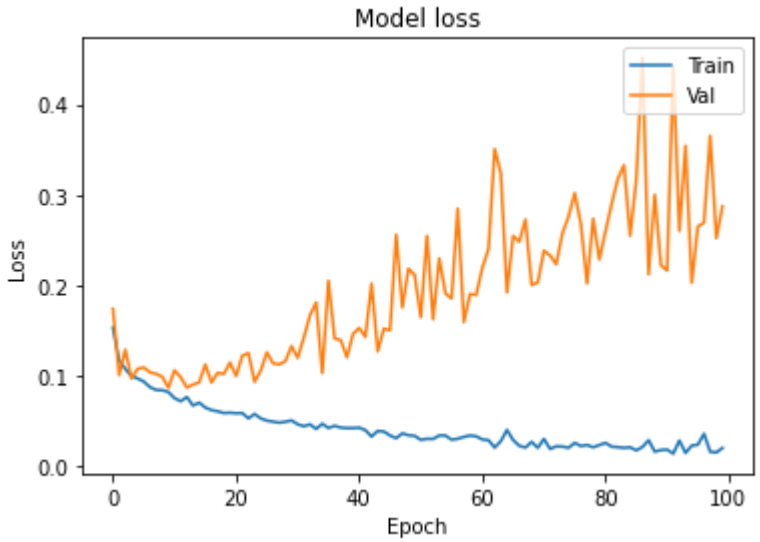
In [171]:

```
model_2 = Sequential([
    Dense(1000, activation='relu', input_shape=(15,)),
    Dense(1000, activation='relu'),
    Dense(1000, activation='relu'),
    Dense(1000, activation='relu'),
    Dense(1, activation='sigmoid'),
])

model_2.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

hist_2 = model_2.fit(x_reduced_trainStand, reduced_y_train,
          batch_size=32, epochs=100,
          validation_data=(x_valStand, y_val))
```

```
Epoch 83/100
270/270 [==============================] - 9s 33ms/step - loss: 0.0210
- accuracy: 0.9918 - val_loss: 0.3178 - val_accuracy: 0.9632
Epoch 84/100
270/270 [==============================] - 9s 34ms/step - loss: 0.0203
- accuracy: 0.9922 - val_loss: 0.3329 - val_accuracy: 0.9616
Epoch 85/100
270/270 [==============================] - 9s 34ms/step - loss: 0.0209
- accuracy: 0.9925 - val_loss: 0.2548 - val_accuracy: 0.9600
Epoch 86/100
270/270 [==============================] - 10s 37ms/step - loss: 0.017
5 - accuracy: 0.9947 - val_loss: 0.3169 - val_accuracy: 0.9630
Epoch 87/100
270/270 [==============================] - 11s 40ms/step - loss: 0.021
2 - accuracy: 0.9948 - val_loss: 0.4519 - val_accuracy: 0.9611
Epoch 88/100
270/270 [==============================] - 10s 35ms/step - loss: 0.028
7 - accuracy: 0.9903 - val_loss: 0.2122 - val_accuracy: 0.9657
Epoch 89/100
270/270 [==============================] - 13s 47ms/step - loss: 0.016
```

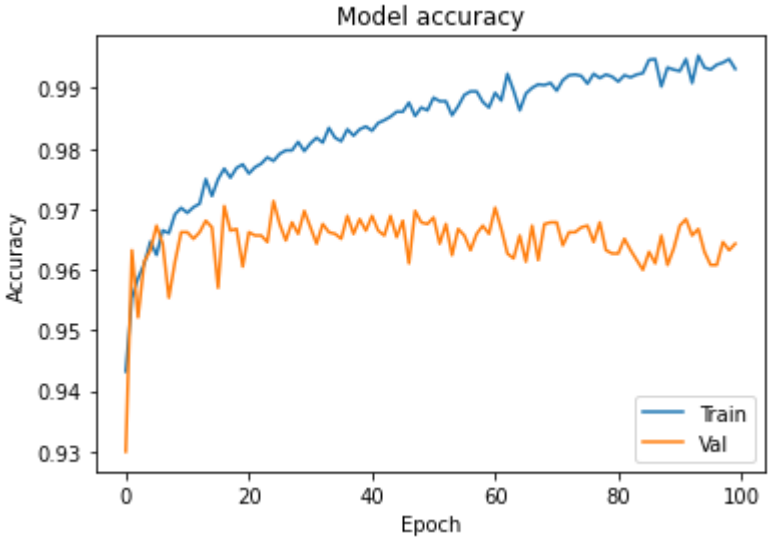
In [176]:

```
#Loss curves for over-fitting model
plt.plot(hist_2.history['loss'])
plt.plot(hist_2.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.show()
```



In [175]:

```
#Training and validation accuracy for our overfitting model
plt.plot(hist_2.history['accuracy'])
plt.plot(hist_2.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='lower right')
2plt.show()
```



In [177]:

```
#strategies to reduce over-fitting
from keras.layers import Dropout
from keras import regularizers
```

In [178]:

```
model_3 = Sequential([
    Dense(1000, activation='relu', kernel_regularizer=regularizers.l2(0.01), input_s
    Dropout(0.3),
    Dense(1000, activation='relu', kernel_regularizer=regularizers.l2(0.01)),
    Dropout(0.3),
    Dense(1000, activation='relu', kernel_regularizer=regularizers.l2(0.01)),
    Dropout(0.3),
    Dense(1000, activation='relu', kernel_regularizer=regularizers.l2(0.01)),
    Dropout(0.3),
    Dense(1, activation='sigmoid', kernel_regularizer=regularizers.l2(0.01)),
])
```

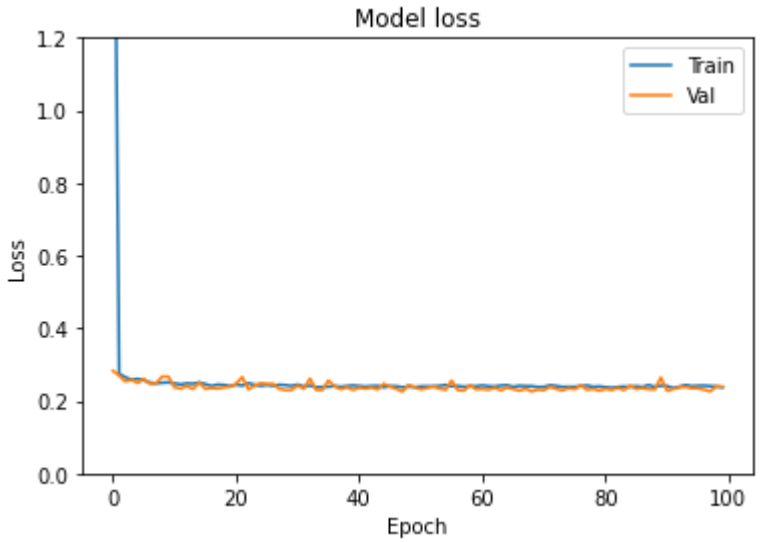
In [179]:

```
model_3.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

hist_3 = model_3.fit(x_reduced_trainStand, reduced_y_train,
          batch_size=32, epochs=100,
          validation_data=(x_valStand, y_val))
```

```
270/270 [==============================] - 16s 58ms/step - loss: 0.245
1 - accuracy: 0.9471 - val_loss: 0.2340 - val_accuracy: 0.9521
Epoch 13/100
270/270 [==============================] - 13s 48ms/step - loss: 0.248
1 - accuracy: 0.9446 - val_loss: 0.2408 - val_accuracy: 0.9532
Epoch 14/100
270/270 [==============================] - 14s 53ms/step - loss: 0.247
1 - accuracy: 0.9482 - val_loss: 0.2327 - val_accuracy: 0.9546
Epoch 15/100
270/270 [==============================] - 14s 51ms/step - loss: 0.250
5 - accuracy: 0.9438 - val_loss: 0.2508 - val_accuracy: 0.9481
Epoch 16/100
270/270 [==============================] - 14s 50ms/step - loss: 0.246
1 - accuracy: 0.9446 - val_loss: 0.2330 - val_accuracy: 0.9505
Epoch 17/100
270/270 [==============================] - 13s 47ms/step - loss: 0.240
8 - accuracy: 0.9484 - val_loss: 0.2353 - val_accuracy: 0.9540
Epoch 18/100
270/270 [==============================] - 15s 55ms/step - loss: 0.244
9 - accuracy: 0.9466 - val_loss: 0.2338 - val_accuracy: 0.9497
```

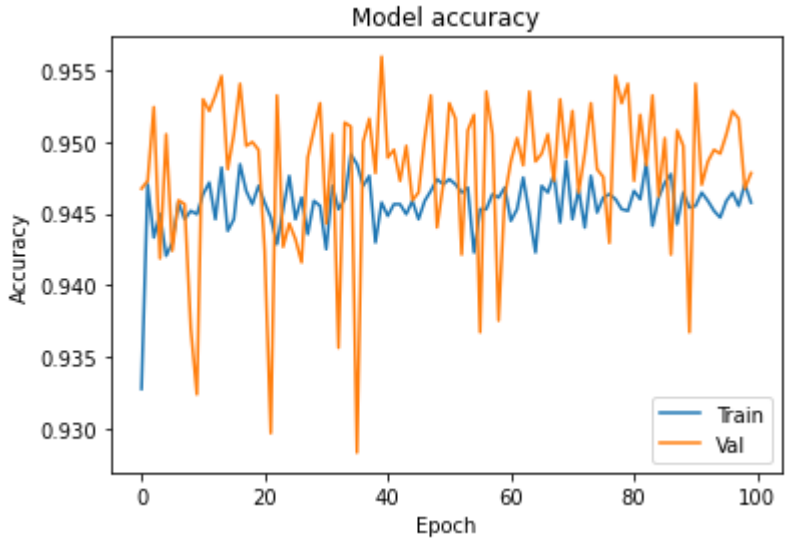
In [180]:

```
plt.plot(hist_3.history['loss'])
plt.plot(hist_3.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.ylim(top=1.2, bottom=0)
plt.show()
```



In [181]:

```
plt.plot(hist_3.history['accuracy'])
plt.plot(hist_3.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='lower right')
plt.show()
```



In [182]:

```
model.evaluate(x_valStand, y_val)[1]
```

```
116/116 [==============================] - 1s 4ms/step - loss: 0.0923
- accuracy: 0.9619
```

Out[182]:

```
0.9618712663650513
```

In [184]:

```
model_3.evaluate(x_valStand, y_val)[1]
```

```
116/116 [==============================] - 2s 16ms/step - loss: 0.2389
- accuracy: 0.9478
```

Out[184]:

```
0.947809636592865
```

In [185]:

```
model_2.evaluate(x_valStand, y_val)[1]
```

```
116/116 [==============================] - 2s 13ms/step - loss: 0.2875
- accuracy: 0.9643
```

Out[185]:

```
0.96430504322052
```

HIGHEST ACCURATE MODEL : RANDOM FOREST